Universidad Carlos III de Madrid
www.uc3m.es

Trabajo Fin de Grado:

METODOLOGÍA DE EMULACIÓN DE ATAQUES EN SISTEMAS DE FIRMA MANUSCRITA

AUTOR: José Antonio Amores Durán.

TUTOR: Raúl Sánchez Reíllo.

Leganés, 24 de Junio de 2013

AGRADECIMIENTOS

En este apartado quiero hacer mención a mis padres, José Antonio y Mª del Carmen, y a mi hermano, Alberto, por su apoyo y confianza.

Recordar también al resto de mi familia que ha estado siempre a mi lado y me ha prestado su apoyo en todo momento y quisiera hacer una mención especial a mi abuelo Félix quien me indicó la referencia *"Eres maestro de lo que has vivido, artesano de lo que estás viviendo y aprendiz de lo que vivirás" de Richard Bach* y que tengo tan presente hoy en dia.

Por último quiero agradecer su apoyo a mi tutor y a mis compañeros del GUTI que tanto me han ayudado ya que sin ellos no habría podido terminar el presente proyecto.

Gracias a todos.

RESUMEN

El objetivo del presente proyecto es realizar una metodología de emulación de ataques en sistemas biométricos mediante firma manuscrita dinámica. Para ello, el proyecto se divide en diferentes fases que finalmente se integrarán en una única aplicación:

1) Crear aplicación perito grafólogo, que contengan funciones de play, forward, reverse, cursor de desplazamiento, hacer zoom y sacar gráficas dinámicamente.

2) Crear una plataforma de emulación de ataques, para diversas versiones: Ciega, Visual, Calcar, Reproducir en tiempo real, Reproducir mediante aplicación perito.

3) Crear una base de datos de firmas falsas

4) La creación de una aplicación para reclutar firmas de usuarios legítimos.

ABSTRACT

The objective of this project is the attacks emulation in the handwritten signature. For this reason, the project is divided into different phases that will eventually be integrated into a single application:

1) Create Application handwriting expert, containing functions of play, forward, reverse, shift cursor, zoom and draw graphics dynamically.

2) Create a platform emulation attacks for various versions: Blind, Visual, Tracing, realtime Play, Play by applying expert.

3) Create a fake signatures database.

4) The implementation of an application for recruiting signatures legitimate users.

Índice de contenido

Índice de figuras

Índice de tablas

Índice de acrónimos

[1] BBDD: Base de Datos

1 Introducción

Éste capítulo describe de una forma general el tema sobre el que trata el presente proyecto introduciendo de este modo el resto del documento. En este punto podemos encontrar los objetivos y motivaciones que han llevado a realizar el mismo. Del mismo modo se ha realizado una breve descripción del resto de contenidos.

1.1 Antecedentes

En este capítulo expondremos los antecedentes existentes sobre el estudio de vulnerabilidades de firma manuscrita, así como los estudios que se han realizado sobre los falsificadores. Hay que destacar que nos hemos basado en esta información para diseñar la aplicación. Lo que se explica a continuación se ha obtenido de la norma ISO/IEC JTC 1/SC 37 N 2239 BIOMETRICS [14].

Para realizar un estudio correcto de la vulnerabilidad de un sistema basado en reconocimiento biométrico tenemos que considerar el esfuerzo que tiene que realizar el falsificador para conseguir realizar la falsificación. Ya que los resultados podrían verse influenciados por el grado de esfuerzo con el que el impostor intenta imitar un comportamiento de un sujeto legítimo. Se sabe que el rendimiento del sistema de autenticación se puede ver influenciado por el grado de esfuerzo del falsificador.
Se pueden describir hasta cuatro tipos de intento de falsificación:

- Falsificación con esfuerzo mínimo: el falsificador presentará sus propias características biométricas y tratará de realizar una verificación correcta en contra de su propia plantilla.

- Falsificación simple: el falsificador tendrá delante alguna característica biométrica del usuario a falsificar, por ejemplo el trazado de su firma auténtica.

- Falsificación simulada: el falsificador copia las características biométricas originales de la firma que está falsificando.

- Falsificación en nivel experto: el falsificador imita la información estática y dinámca de las características biométricas de la firma legítima, a menudo con la observación y la práctica.

Es razonable pensar que el nivel de falsificador con esfuerzo mínimo obtendrá unos valores de falsificación menores que los del resto de falsificadores ya que está “falsificando” su propia firma.

Por lo tanto, es necesario tener el grado de esfuerzos del impostor en consideración para obtener resultados analíticos fiables en las modalidades de comportamiento.

Una forma de tomar en cosideracion el grado de esfuerzos del falsificador en las pruebas de autenticación basadas en firmas es el uso de la categorización de los tipos de falsificación que se exponen a continuación:

Nivel	Descripción	Tipos
0	Falsificación con esfuerzo cero, el falsifcador firma un nombre aleatorio (no conoce ningún dato de quien va a falsificar).	Falsificaciones a "ciegas", el falsificador no tiene acceso a la firma del usuario que quiere falsificar.
1	El falsificador ha oido el nombre pero no ha visto la firma de quien va a falsificar, ni el nobre escrito (Ej. Steven vs. Stephen).	
2	El falsificador ha visto escrito el nombre del usuario al que va a tratar de falsificar pero no conoce su firma.	
3	El falsificador tiene acceso a una única muestra del usuario que quiere falsificar.	Falsificaciones estáticas, el falsificador únicamente tiene acceso a la "imagen" de la firma (posiciones x e y en el plano) pero no se le muestra el progreso, no puede ver como se ha realizado.
4	El falsificador tiene acceso a múltiples muestras de la firma del usuario a falsificar.	
5	El falsificador ha observado al usuario realizar su firma.	Falsificaciones basadas en observación, el falsificador ha visualizado a la "víctima" realizar su firma.
6	Observación múltiple: el falsifiador ha observado en varias ocasiones la realización de la firma del usuario qu quiere falsificar.	
7	El usuario a falsificar instruye al falsificador a realizar su firma.	Falsificación asistida, ya sea a través del propio usuario legítimo o mediante tecnología.
8	"Tecnología de asistencia", el falsificador tiene acceso a los datos completos de la firma y se le permite prácticar múltiples intentos de imitar la velocidad, la presión y las curvas.	

Tabla 1 Tabla de niveles de la norma

1.2 Descripción general

Actualmente el reconocimiento de una persona mediante técnicas biométricas está tomando cada vez más importancia. Estas técnicas biométricas tratan de identificar a una persona mediante unos rasgos físicos o de comportamiento. Una de las técnicas de reconocimiento biométrico (basada en rasgos de comportamiento) más extendidas es la firma manuscrita, en este caso se trata de identificar al sujeto a través de su forma de escribir, concretamente se estudia su símbolo identificativo.

El hecho de incorporar sistemas de identificación biométrica basados en firma manuscrita tiene como inconveniente que al ser una técnica basada en el comportamiento puede ser susceptible de fraude.

Debido a este problema se pretende estudiar el grado de parecido que puede llegar a tener la falsificación de una firma de un usuario genuino en función del conocimiento que tenga el falsificador de dicha firma.

Se ha realizado este proyecto con el fin de poder realizar este estudio mediante una aplicación que muestre en diferentes niveles distintos parámetros de la firma del usuario genuino con el fin de que el falsificador conozca en cada nivel un parámetro distinto de la firma del usuario que va a falsificar (se mostrará la imagen de la firma, el progreso de la misma, la velocidad del trazo, etc.), llegando a conocer en los últimos niveles la firma con todo detalle. De esta forma se podrá estudiar el parecido de la firma falsificada y la firma genuina en función del conocimiento del falsificador sobre dicha firma.

Para capturar los datos necesarios para realizar dicha evaluación se ha utilizado la tableta STU-WACOM la cual es capaz de capturar diversos parámetros de la firma como son la posición (x, y), la presión y el tiempo.

La aplicación consta de siete niveles de falsificación que se basan en los niveles de la norma ISO/IEC JTC 1/SC 37 N 2239 [14] y que permiten evaluar el esfuerzo del falsificador en realizar las falsificaciones de la firma legitima en funcion de su conocimiento sobre la misma. De esta forma se puede realizar el sistema de emulación de ataques en firma manuscrita que es la base de la aplicación.

La aplicación permite trabajar en dos modos distintos:

- Reclutamiento de usuarios genuinos: este modo se reclutarán las firmas legítimas de usuarios reales. En este modo el usuario que realice su firma deberá dar sus datos personales (DNI, su lateralidad, etc.) y se le asignará un número de usuario en el sistema.

- Reclutamiento de usuarios falsificadores: en este modo se reclutarán las falsificaciones de los usuarios falsificadores que intentarán falsificar las firmas de los usuarios genuinos a través de diferentes niveles. Del mismo modo que anteriormente será necesario un registro de estos usuarios (DNI, nombre, etc.) y se le asignará un numero de usuario en el sistema.

La aplicación desarrollada ha recibido el nombre de "Handwritten Signature Vulnerabilities Toolbox" y se utilizará como herramienta de apoyo a la investigación.

1.3 Objetivos

El presente proyecto tiene como objetivo principal desarrollar una aplicación para Windows que permita mostrar en diferentes niveles las características que definen la firma legítima de un usuario. Esta aplicación tiene también como objetivo la adquisición de las muestras de firma legitimas de usuarios así como la adquisición de muestras de las firmas falsificadas en función del conocimiento que el falsificador haya adquirido al ver las características de la firma genuina.

Más allá de estos objetivos, el fin de la aplicación es conseguir un soporte para estudiar el grado de parecido de una firma realizada por un usuario legítimo y la propia firma falsificada en función de los conocimientos que tenga el falsificador sobre la misma.

1.4 Motivación

Aparte de los objetivos del proyecto es necesario hablar de qué motiva la creación de esta aplicación y lo que se pretende conseguir con ella.
Debido a que el reconocimiento de una persona mediante técnicas biométricas está cada vez más presente en nuestros días surge la necesidad de estudiar el grado de vulnerabilidad que presentan dichas técnicas de reconocimiento. Es por este motivo por el que nace esta aplicación, para realizar un estudio que compare las muestras legítimas de las firmas de un usuario con las muestras falsificadas de las mismas en función del grado de conocimiento que tenga el falsificador sobre las características de las firmas que quiere imitar.

Resulta complicado realizar dicho estudio comparativo sin una aplicación que facilite la adquisición de muestras con los datos necesarios de las firmas.

Con los datos que se obtengan se podrá llevar a cabo un estudio comparativo entre las firmas legítimas y las falsificadas. La motivación de esta aplicación es servir de soporte para conseguir los archivos necesarios para llevar a cabo este estudio comparativo.

1.5 Etapas de desarrollo

En primer lugar se llevó a cabo una labor de documentación sobre el entorno de desarrollo de Microsoft Visual Studio ya que es el entorno utilizado para realizar la aplicación que se describe en el proyecto. Se recopiló información sobre diversos aspectos de la programación de aplicaciones en este entorno (lenguaje, interfaz con el usuario y demás aspectos). Fue necesario buscar documentación sobre la programación orientada a objetos, la aplicación ha sido programada en el lenguaje de programación C#.

El siguiente paso fue la recopilación de información sobre la técnica biométrica de firma manuscrita ya que es el sistema electrónico de identificación en el que se basa el proyecto. Es necesario estudiar las principales características de esta técnica de reconocimiento para conocer los parámetros relevantes que tenemos que tener en cuenta a la hora de guardar los datos que son importantes para definir la firma de una persona.

Posteriormente se seleccionó un dispositivo capaz de obtener los parámetros necesarios que definen la firma de una persona. Se seleccionó la tableta STU-WACOM-500 que es capaz de registrar parámetros como la presión que ejerce el usuario al firmar, el tiempo que tarda en realizar dicha firma, etc. ya que es una tableta especializada en la captura de firmas digitales.

Finalmente se definió la aplicación a desarrollar, esta aplicación debía tener una fácil interacción entre el usuario y el sistema, al definirla se pretendía que fuera intuitiva y fácil de utilizar pero que a la vez cumpliera con sus objetivos de forma eficiente. El idioma elegido para la aplicación fue el inglés, ya que esta aplicación se plantea usar para fines investigadores donde este lenguaje es el más empleado.

1.6 Resto del documento

En el resto del documento se describe la arquitectura utilizada para desarrollar la aplicación, el diseño que finalmente se ha escogido para la misma y como se ha desarrollado, explicando en cada momento los problemas que han ido surgiendo en este proceso y las soluciones que hemos adoptado.

Se describen también las diferentes pruebas a las que se sometió la aplicación para comprobar que funciona correctamente y que la usabilidad es correcta.

También describimos los cambios realizados debido a los fallos que surgieron en dichas pruebas.

Así mismo explicamos la base de datos realizada con firmas legítimas, el número de usuarios, las características de cada uno, etc.

Finalmente el presente documento finaliza con las conclusiones y líneas futuras del autor sobre la aplicación.

2 Estado del arte

En este capítulo se procederá a introducir la tecnología utilizada para desarrollar la aplicación "Handwritten Signature Vulnerabilities Toolbox".

2.1 Biometría

2.1.1 Definición de biometría y características de un sistema biométrico

Biometría es la ciencia por la que se puede identificar a una persona basándose en sus características físicas o de comportamiento [3].

Se pretende automatizar una tarea que el cerebro humano realiza con facilidad, los dos objetivos principales de la biometría son:

- Realizar la identificación en entornos donde no se puede tener a una persona para realizar dicha identificación [3].

- Evitar el riesgo de fallos por cansancio humano [3].

La biometría empieza a surgir como ciencia cuando se necesita constatar objetivamente la identidad de una persona [3].

La firma manuscrita se puede considerar la primera técnica biométrica utilizada (para aquellas personas que no sabían escribir se les pedía que hicieran una cruz o una marca característica) [3].

Todos los sistemas de identificación biométrica se basan en dos fases diferenciadas [3]:

- Reclutamiento: todos los usuarios que van a formar parte del sistema han de pasar por un centro en el que se tomen sus datos.
  El objetivo será crear un patrón que sea característico de cada usuario.
  El número de muestras que se han de tomar del sujeto para obtener el patrón dependerá de la técnica y de la configuración del sistema.

- Identificación: cuando se debe introducir la identificación de un sujeto se tomará una muestra del mismo y el vector de características resultante se comparará con el patrón previamente almacenado.

El proceso de pasar de un sujeto a un vector de características se puede dividir en diferentes pasos (Ilustración 1) [3][2]:

- Captura/ toma de muestras: en el caso de este proyecto para la captura de muestras se utilizará una tableta STU-WACOM-500

- Pre-procesado: es necesario eliminar la información no útil de las muestras que se han recogido en el proceso de captura de muestras. También es necesario adaptar la información útil para pasar al siguiente paso.

- Extracción de características: analiza la información que le llega, y la procesa extrayendo los parámetros propios del sujeto. Finalmente del resultado de este último paso se obtiene el vector de características.

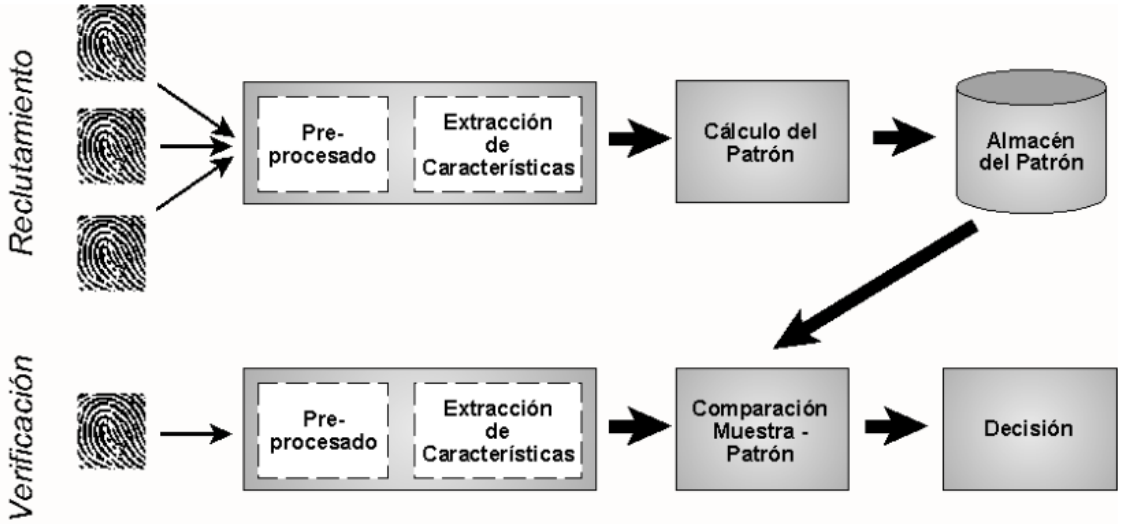

Ilustración 1. Reclutamiento y verificación de muestras **[3]**

En un sistema biométrico el reconocimiento de un sujeto trata de identificar a una persona dentro de un grupo previamente reclutado y por consiguiente podemos obtener los siguientes resultados [3]:

- Podemos obtener el usuario del grupo que más se parece.
- El usuario del grupo que más se parece con una tasa de parecido.
- Un rechazo si el parecido no supera el umbral o una aceptación con el usuario especificado.

En la autenticación de un sujeto se compara la muestra obtenida del usuario con el patrón almacenado de esa supuesta misma persona.

Es necesario medir el rendimiento de un sistema biométrico, para ello se establecen tres parámetros típicos [3]:

- Tasa de falso rechazo: esta tasa mide la cantidad de veces que un usuario válido es rechazado.
- Tasa de falsa aceptación: porcentaje de veces en las que un usuario no válido es aceptado.
- Tasa de igual error: punto en el que la tasa de falso rechazo y la tasa de falsa aceptación tienen el mismo valor.

Hay que tener en cuenta que los usuarios pueden rechazar un sistema biométrico por múltiples razones [3]:

- Temor a sufrir daños colaterales, ya sea por desconocimiento del funcionamiento del sistema en el caso de identificación de iris.
- Por implicaciones legales en el caso de que el usuario tenga que dejar su muestra de huella dactilar.
- Por falta de comodidad.
- Por desidia o cansancio.
- Por tasas de error elevadas.

Por estos motivos es importante realizar las aplicaciones lo más intuitivas y sencillas para los usuarios, es importante que sean efectivas y que tengan bajas tasas de error, también es muy importante informar al usuario de la tecnología que se va a utilizar.

2.1.2 Parámetros que determinan si un rasgo físico puede usarse con fines biométricos

A la hora de juzgar un sistema biométrico tenemos que basarnos en diferentes parámetros, podemos destacar los siguientes [3]:

- Universalidad: si las características se pueden extraer de cualquier usuario o no.

- Unicidad: la probabilidad de que no existan dos usuarios con las mismas características.

- Estabilidad: si las características que se extraen permanecen inalterables en relación con diversos parámetros como puede ser la edad, enfermedades en el usuario, etc.

- Facilidad para captura: si existen mecanismos sencillos de captura de datos biológicos o de comportamiento del sujeto.

- Rendimiento o tasas de acierto y error.

- Aceptación por los usuarios.

- Robustez frente a la burla del sistema: si la técnica puede reconocer el falseamiento de los datos capturados.

- Coste.

2.1.3 Identificación biométrica mediante firma manuscrita

La identificación biométrica mediante la firma manuscrita es una técnica más antigua que la huella dactilar, esta técnica siempre se ha visto entredicha por la posibilidad de ser falsificada con relativa facilidad debido a que está basada en características del comportamiento [5].

La firma es el primer símbolo que se utilizó para certificar la presencia/aceptación/rechazo de una persona, es la única técnica reconocida por todo el mundo como muestra de confirmación voluntaria y legal de un usuario [5].
Hoy en día se encuentra totalmente aceptada y se utiliza en multitud de entornos.

Realizar experimentos serios de esta técnica implica que se cuente con falsificadores profesionales lo que puede ser una tarea con poca expectativa de éxito, además cualquier persona puede intentar falsificar la firma de otra, pero la calidad de la misma es muy inferior a la de un profesional.

Existen dos filosofías [5]:

- Firma estática (conocida como off-line) en la que se trabaja con una imagen de la firma y se estudia su grafología. Esta técnica se basa en el procesado de imagen, el método de captura se realiza mediante un escaneado de la imagen de la firma.

- Firma dinámica (conocida como on-line) en esta técnica no solo se captura la imagen sino los distintos parámetros relacionados con el acto de firmar, la captura se realiza mediante una tableta digitalizadora de altas prestaciones, se capturan parámetros como la presión, el movimiento en el eje X, el movimiento en el eje Y, el ángulo de inclinación del bolígrafo, etc.

Tanto la filosofía de firma estática como la firma dinámica se han evaluado e integrado en la presente aplicación, en los posteriores capítulos se explicará su integración en la misma.

Actualmente las nuevas tecnologías permiten el estudio de la firma y además permiten el estudio del acto de firmar captando mediante un bolígrafo especial o una tableta gráfica parámetros como velocidad, paradas, posición del bolígrafo, etc. en el mismo acto de firmar.

Al igual que todas las técnicas biométricas en el caso de la identificación de firma manuscrita encontramos puntos a favor y puntos en contra [3]. Analizaremos los siguientes parámetros a favor y en contra de la identificación biométrica a través de la firma ([3]Tabla 2):

Técnica de identificación biométrica: firma manuscrita	
A favor	**En contra**
Fácil de capturar	Universalidad baja en diversos entornos
Aceptado por los usuarios (ya que están acostumbrados a firmar en muchos entornos)	Unicidad no muy elevada
Soporte legal	Estabilidad baja
	Coste no muy bajo. Si es dinámica el bolígrafo o la tableta es costosa y con susceptibilidad de extravío
	No es cómodo en determinados entornos
	Susceptible de fraude

[3]Tabla 2. Evaluación de la técnica de firma manuscrita como técnica de identificación biométrica

2.2 Dispositivos de captura de firmas

Hoy en día existen multitud de dispositivos táctiles que permiten capturar y digitalizar la firma de un usuario, entre estos dispositivos podemos encontrar teléfonos móviles, tabletas, tabletas especificas de captura de firmas (como la utilizada), pantallas de ordenadores táctiles, y dispositivos similares. A continuación describiremos algunos de estos dispositivos:

2.2.1 Tabletas digitalizadoras de firmas:

En el mercado podemos encontrar diversas marcas de tabletas digitalizadoras de firmas pero destacamos ante todas la STU-WACOM [6], podemos encontrar diversos modelos de este tipo de tabletas (STU-520, STU-500 y STU-300). Son dispositivos específicos para la captura de firmas, por lo que disponen de una superficie cómoda para firmar que simula la sensación de la firma en papel y son perfectos para transacciones orientadas al usuario, porque la firma es visible mientras se escribe en la pantalla.

Los lápices son inalámbricos, sin baterías y no requieren mantenimiento.

El sensor registra la información biométrica pertinente de la firma: como los datos coordinados y la sensibilidad a la presión.

2.2.2 Teléfonos móviles con pantalla táctil:

En la actualidad muchos de los teléfonos móviles incorporan pantallas táctiles que podrían utilizarse para la captura digital de la firma de un usuario [13].

2.2.3 Tablets:

Hoy en día existen multitud de tabletas que tienen una pantalla de gran tamaño y táctil, lo que nos lleva a pensar que también podían ser utilizadas para la captura de firma digital [13].

2.2.4 Otros dispositivos con pantalla táctil:

En este apartado podemos clasificar los ordenadores con pantalla táctil y los terminales con pantalla táctil que muchas veces nos encontramos en las tiendas a la hora de realizar un pago, estos dispositivos se utilizan en muchas ocasiones para capturar firmas [13].

3 Programación orientada a objetos

El lenguaje de programación escogido para desarrollar la aplicación es C#, que es un lenguaje de programación orientado a objetos, a continuación daremos una breve introducción sobre este tipo de programación.

La programación orientada a objetos ofrece mejoras en la forma de diseño, desarrollo y mantenimiento del software, de esta forma se consigue una solución a largo plazo de los problemas que han existido desde el comienzo en el desarrollo del software: la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas [12].

El lenguaje de programación orientado a objetos tiene que tener las siguientes características: debe estar basado en objetos, basado en clases y ser capaz de tener herencia de clases [12].

El elemento fundamental en este tipo de programación es como su nombre indica el objeto. El objeto se puede definir como un conjunto complejo de datos y programas que poseen estructura y forma parte de una organización.

Un objeto incluye en su interior un conjunto de componentes bien estructurados y además forma parte de una organización jerárquica.

La estructura de un objeto se divide en tres partes [12]:

- Relaciones: permiten que el objeto se inserte en la organización y están formadas por punteros a otros objetos.

- Propiedades: distinguen a un objeto determinado de los restantes que forman parte de la misma organización y tiene valores que dependen de la propiedad de que se trate. Las propiedades de un objeto pueden ser heredadas a sus descendientes en la organización.

- Los métodos: son las operaciones que pueden realizarse sobre el objeto, que normalmente estarán incorporados en forma de programas (código) que el objeto es capaz de ejecutar y que también pone a disposición de sus descendientes a través de la herencia.

Es necesario destacar los siguientes conceptos sobre este tipo de programación [12]:

- Clase: definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas.

- Evento: es un suceso en el sistema (tal como una interacción del usuario con la máquina, o un mensaje enviado por un objeto). El sistema maneja el evento enviando el mensaje adecuado al objeto pertinente. También se puede definir como evento la reacción que puede desencadenar un objeto; es decir, la acción que genera.

- Atributos: características que tiene la clase.

- Estado interno: es una variable que se declara privada, que puede ser únicamente accedida y alterada por un método del objeto, y que se utiliza para indicar distintas situaciones posibles para el objeto (o clase de objetos). No es visible al programador que maneja una instancia de la clase.

Las características que contempla la programación orientada a objetos son las siguientes [12]:

- Abstracción: denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos, y, cuando lo están, una variedad de técnicas son requeridas para ampliar una abstracción. El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real. La abstracción es clave en el proceso de análisis y diseño orientado a objetos, ya que mediante ella podemos llegar a armar un conjunto de clases que permitan modelar la realidad o el problema que se quiere atacar.

- Encapsulamiento: significa reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema.

- Modularidad: se denomina modularidad a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos se pueden compilar por separado, pero tienen conexiones con otros módulos. Al igual que la encapsulación, los lenguajes soportan la modularidad de diversas formas.

- Principio de ocultación: cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que específica

cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas; solamente los propios métodos internos del objeto pueden acceder a su estado. Esto asegura que otros objetos no puedan cambiar el estado interno de un objeto de manera inesperada, eliminando efectos secundarios e interacciones inesperadas

- Polimorfismo: comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre; al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O, dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.

- Herencia: las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento, permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases y estas en árboles o enrejados que reflejan un comportamiento común. Cuando un objeto hereda de más de una clase se dice que hay herencia múltiple.

- Liberación de memoria**:** es la técnica por la cual el entorno de objetos se encarga de destruir automáticamente, y por tanto desvincular la memoria asociada, los objetos que hayan quedado sin ninguna referencia a ellos. Esto significa que el programador no debe preocuparse por la asignación o liberación de memoria, ya que el entorno la asignará al crear un nuevo objeto y la liberará cuando nadie lo esté usando.

3.1. Introducción al lenguaje C#

Debido a que la aplicación se ha desarrollado en este lenguaje de programación a continuación daremos una breve introducción sobre el mismo:

C# es un lenguaje orientado a objetos que permite a los desarrolladores compilar diversas aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Se puede utilizar C# para crear aplicaciones cliente de Windows tradicionales, servicios Web

XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y más aplicaciones. Visual C# proporciona un editor de código avanzado, cómodos diseñadores de interfaz de usuario, un depurador integrado, y otras herramientas para facilitar el desarrollo de aplicaciones basadas en el lenguaje C# [15].

Como lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. Una clase puede heredar directamente de una clase primaria, pero puede implementar cualquier número de interfaces [15].

4 Plataforma de desarrollo

Debido al sensor de captura de firmas escogido y que la aplicación se va a desarrollar para PC se decidió realizar esta aplicación para Windows ya que es el sistema operativo más común y utilizado actualmente.

Como entorno de desarrollo de nuestra aplicación se escogió Microsoft Visual Studio que es un entorno de desarrollo integrado para sistemas operativos Windows. Este entorno de desarrollo soporta varios lenguajes de programación como Visual C++, Visual C# (que es el que hemos utilizado), Visual J# y Visual Basic .NET, al igual que entornos de desarrollo web como ASP .NET.
Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web.

4.1. Introducción a Visual Studio

Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos el mismo entorno de desarrollo integrado (IDE), que habilita el uso compartido de herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones web ASP y Servicios Web XML [1][10].

4.2. Desarrollo de aplicaciones en Visual C#

Nuestra aplicación se ha desarrollado mediante el entorno de desarrollo integrado de Visual C#, concretamente es una aplicación de Windows Forms, que es un conjunto de herramientas de desarrollo. Algunas de estas herramientas se comparten con otros lenguajes de programación de Visual Studio y otras como el compilador de C# son exclusivas de Visual C# [10].

4.3. Arquitectura de la plataforma .NET Framework

Los programas de C# se ejecutan en .NET Framework, un componente que forma parte de Windows y que incluye un sistema de ejecución virtual denominado Common Language Runtime (CLR) y un conjunto unificado de bibliotecas de clases. CLR es la implementación comercial de Microsoft de CLI (Common Language Infrastructure), un estándar internacional que constituye la base para crear entornos de ejecución y desarrollo en los que los lenguajes y las bibliotecas trabajan juntos sin ningún problema [7].

El código fuente escrito en C# se compila en un lenguaje intermedio (IL) conforme con la especificación CLI. El código de lenguaje intermedio y recursos tales como mapas de bits y cadenas se almacenan en disco en un archivo ejecutable denominado ensamblado, cuya extensión es .exe o .dll generalmente. Un ensamblado contiene un manifiesto que proporciona información sobre los tipos, la versión, la referencia cultural y los requisitos de seguridad del ensamblado.

Cuando se ejecuta un programa de C#, el ensamblado se carga en CLR, con lo que se pueden realizar diversas acciones en función de la información del manifiesto. A continuación, si se cumplen los requisitos de seguridad, CLR realiza una compilación Just In Time (JIT) para convertir el código de lenguaje intermedio en instrucciones máquina nativa. CLR también proporciona otros servicios relacionados con la recolección de elementos no utilizados automática, el control de excepciones y la administración de recursos. El código ejecutado por CLR se denomina algunas veces "código administrado", en contraposición al "código no administrado" que se compila en lenguaje máquina nativo destinado a un sistema específico. En el diagrama siguiente se muestran las relaciones en tiempo de compilación y tiempo de ejecución de los archivos de código fuente de C#, las bibliotecas de clases de .NET Framework, los ensamblados y CLR (Ilustración 2) [7].

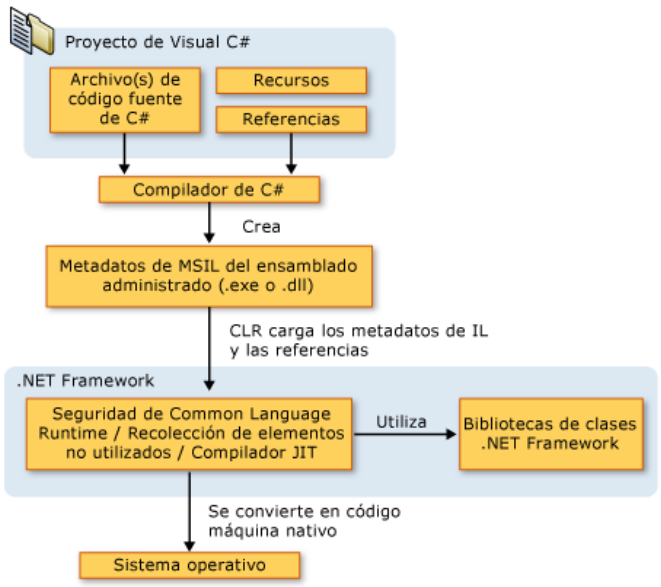

Ilustración 2. Arquitectura de la plataforma .NET Framework [7]

5 Análisis y diseño.

En este capítulo hablaremos sobre los requisitos que requiere la aplicación y que han condicionado el diseño de la misma:

Esta aplicación ha sido desarrollada para Windows como hemos explicado anteriormente, se ha escogido este sistema operativo en relación a los dispositivos que utilizamos y por ser uno de los más utilizados y universales de nuestros tiempos.

Por tanto será requisito necesario para que la aplicación funcione un PC con SO Windows, también deberá disponer de al menos un puerto USB ya que necesitaremos conectar la tableta STU-WACOM. Será necesario disponer de espacio en la memoria suficiente para guardar las muestras que reclutemos.

5.1. Análisis de la aplicación

La aplicación desarrollada se basa en el análisis de la metodología de emulación de ataques en sistemas de firma manuscrita y en la captura de firmas legítimas.
Para realizar la aplicación que emula ataques en sistemas de firma manuscrita y obtener las muestras que posteriormente permitirán estudiar la vulnerabilidad de este sistema de reconocimiento biométrico nos hemos basado en los niveles que describe la norma ISO/IEC JTC 1/SC 37 N 2239 [14] y que se han explicado detalladamente en capítulo de Introducción concretamente en Antecedentes. A partir de estos niveles se ha diseñado la aplicación presente, que se divide en un total de siete niveles de falsificación que nos permitirán tener en consideración el esfuerzo que tiene que realizar el falsificador para conseguir realizar la falsificación. A continuación explicamos los siete niveles detalladamente:

- Nivel 1: en el nivel 1 no se le mostrará ninguna información sobre la firma que se va a falsificar, es decir, en este nivel se realizará una falsificación “a ciegas”. El falsificador no conoce ninguna característica de la firma. Este nivel en la aplicación sería el equivalente al nivel cero de la norma.

- Nivel 2: en este nivel se le mostrará al usuario durante un breve periodo de tiempo (aproximadamente 5s) la imagen de la firma del usuario que está falsificando. El falsificador adquirirá conocimientos sobre la firma estática del usuario que está falsificando. Transcurridos estos 5s la imagen de la firma desaparecerá. El falsificador podrá comenzar a firmar desde el primer momento, es decir no tendrá por qué esperar a que la imagen de la firma desaparezca para comenzar a realizar las muestras falsas. El diseño de este nivel se basa en el nivel 3 de la norma.

- Nivel 3: en este nivel se le mostrará al falsificador de forma permanente la imagen de la firma del usuario que está falsificando, es decir dispondrá de forma permanente de la firma estática de la "víctima". Al igual que antes el falsificador podrá comenzar a realizar las muestras falsas en el momento que lo desee.

- Nivel 4: en este nivel se mostrará la imagen de la firma del usuario genuino en la pantalla de la tableta STU-WACOM permitiendo al falsificador que la calque. Ya no se mostrará la imagen en la pantalla del ordenador.

- Nivel 5: en este nivel ya no aparecerá la imagen de la firma en la pantalla de la STU-WACOM. Se mostrará de forma temporal la firma dinámica del usuario que se está falsificando, es decir, se mostrará el progreso de la firma, grosor y demás parámetros característicos. Toda esta información se mostrará en la pantalla del ordenador. No se podrá repetir la visualización de la firma, solo se mostrará una vez.

- Nivel 6: en nivel 6 el falsificador podrá hacer uso del "Reproductor de firmas". En este nivel el falsificador tendrá un control total sobre la firma dinámica que está falsificando. Podrá volver a ver el progreso tantas veces como quiera, a una mayor y menor velocidad de reproducción. Podrá pausar en cualquier momento el progreso y continuar desde ese punto. También podrá realizar las muestras en el momento en el que él lo desee.

- Nivel 7: este es el último nivel de la aplicación en el que se mezclarán el nivel 4 (calco de la firma genuina) y el nivel 6 que hemos descrito anteriormente. En este nivel se le proporciona al usuario falsificador total conocimiento de la firma que está falsificando, se le proporciona tanto la firma estática como la firma dinámica.

Debido a que la aplicación del presente proyecto permite tanto la captura de firmas legítimas como el análisis de la metodología de ataques de firma manuscrita, siendo ambas aplicaciones muy distintas se decidió diseñarlas como aplicaciones separadas dentro de una misma aplicación global.

5.2. Diseño

5.2.1 Requisitos de la aplicación

La aplicación "Handwritten Signature Vulnerability Toolbox" podemos diferenciarla en dos sub-aplicaciones:

- Una aplicación de captura de muestras legítimas.

- Una aplicación de emulación de ataques en firma manuscrita.

Para el primer caso la aplicación debe ser capaz de tomar muestras legítimas de los usuarios que se ofrezcan, por lo tanto esta aplicación será capaz de registrar el nombre del usuario que realiza la firma, asignarle un número de usuario con el que podrá acceder al programa, y tomar las muestras de forma óptima.

Para el caso de que el usuario no recuerde el número de usuario que se le ha asignado será necesario ofrecer la alternativa de la búsqueda por DNI. Las muestras se tienen que poder identificar a través del número de usuario que las ha realizado y través del número de muestra.

Se ha impuesto también como requisito en la aplicación el caso de que un usuario pueda disponer de más de una firma legítima, para este caso la aplicación podrá registrar las diferentes firmas de un mismo usuario asignándole un número de usuario distinto pero compartiendo DNI (de esta forma las podremos identificar). La aplicación comprobará la identidad del usuario en cada caso.

La sub-aplicación dedicada a la emulación de ataques de firma manuscrita precisa de más restricciones ya que necesitamos registrar al falsificador (que también puede estar registrado como usuario genuino) por lo tanto tenemos que asegurarnos de que el falsificador no se "falsifica" a sí mismo, tenemos que tener un registro de que usuarios a falsificado para que no se vuelva a repetir el proceso (ya que conocería su firma después de pasar por todos los niveles y no tendría sentido).

También es importante que el usuario falsificador comience desde el nivel 1 y vaya avanzando progresivamente hasta el nivel 7 según la aplicación le indique, es decir, no se podrán realizar los niveles de forma salteada ya que se conocerían prematuramente características de la firma que no se deberían y por consiguiente el reclutamiento no serviría para la investigación.

Al igual que en la anterior sub-aplicación el sistema de registro se basa en asignar al usuario un número personal y en el caso de que no se recuerde se le permitirá su búsqueda por medio de su DNI. Las muestras de esta sub-aplicación deberán poder identificarse a través de número de falsificador, número de usuario que ha falsificado, nivel utilizado y muestra realizada.

Ante todo se ha tratado de realizar la aplicación de forma que sea intuitiva y fácil de utilizar para el usuario.

Es requisito indispensable para las dos sub-aplicaciones que el usuario se identifique de forma correcta en el sistema para poder avanzar en la aplicación (realizar las falsificaciones, o realizar sus firmas legítimas), se han impuesto estas restricciones

debido a que la aplicación se basa en la recogida de muestras para una futura investigación por lo que es indispensable que estén perfectamente identificadas.

Por último resaltamos que ya que esta aplicación se va a utilizar para fines de investigación se ha decidido implementarla en inglés ya que es el idioma más utilizado.

5.2.2 Elección del dispositivo de captura de firmas

El bloque dedicado a la captura de muestras en un sistema biométrico en muchas ocasiones es "ignorado" pero es de vital importancia para el éxito del sistema. Una de las funciones principales de la aplicación que se describe en el proyecto es el reclutamiento de muestras [4].

La función de este bloque es capturar una muestra de suficiente resolución como para no desvirtuar las características, por este motivo es importante escoger un sensor adecuado para nuestra aplicación [4].

Es importante no limitar la universalidad de las muestras. Además debe ser estable en el tiempo, no debe capturar muestras de peor calidad debido al número de usos o por las condiciones meteorológicas [4].

Tenemos que conseguir que la captura de muestras sea cómoda y natural para el usuario además debe ser atractivo para el mismo. A la hora de realizar la toma de muestras tenemos que intentar no coaccionar el ritmo de vida habitual de los usuarios. Se tiene que intentar que el usuario tenga que intervenir lo menos posible con la aplicación, es decir, la toma de muestras debe ser lo más automatizada posible [4].

El sistema de toma de muestras debe ser de coste y prestaciones coherentes con la seguridad exigida, no tiene sentido tener un sistema de autentificación que sea más caro que lo que tiene que proteger.

Consideraciones sobre el muestreo [4]:

- Al muestrear en el tiempo hay que cubrir con suficiencia los criterios de Nyquist, es decir muestrear al menos al doble de la frecuencia de la máxima que tiene la señal.

- Al muestrear en el espacio hay que aplicar el equivalente del teorema de Nyquist ya que una menor resolución puede provocar pérdida de información y mayor resolución de la necesaria puede provocar un incremento del coste computacional.

- Al cuantificar es necesario asegurarse de que la profundidad de escala es suficiente debido a que el ojo humano no es capaz de reconocer más de 256 niveles de intensidad. Es importante considerar si el color es necesario y que profundidad se necesita.

Todos estos puntos descritos anteriormente se han tenido en cuenta a la hora de escoger el sensor necesario para nuestra aplicación y también para el diseño de la misma.

Esta aplicación requiere un dispositivo con pantalla táctil donde se puedan mostrar las distintas instrucciones para el usuario, que utilizaremos para la adquisición de muestras y que será necesario para llevar a cabo la aplicación de forma satisfactoria.

Anteriormente en el Estado del Arte describimos algunos dispositivos que podíamos encontrar en el mercado que podían permitir la adquisición de firmas digitales, a continuación describiremos los criterios de diseño en los que nos hemos basado para seleccionar uno de ellos e implementarlo en el proyecto.

5.2.2.1 Tabletas digitalizadoras de firmas

Escogimos la tableta STU-WACOM (concretamente se ha trabajado con la tableta STU-WACOM-500) debido a que es un dispositivo específico para la captura de firmas, dispone de una pantalla grande monocromática de 5'' con una resolución VGA de 640 x 480 que permite un amplio espacio para firmar. La pantalla grande ofrece la flexibilidad para mostrar imágenes y texto, o incorporar teclas programables o botones de menú. La tableta STU-WACOM-500 también ofrece la versatilidad de una conexión de datos USB o serial, lo cual es necesario debido a que tenemos que comunicarnos con el PC.

El sensor registra la información biométrica pertinente de la firma: como los datos coordinados y la sensibilidad a la presión.

Como se acaba de describir podemos comprobar que este dispositivo se ajusta perfectamente a las necesidades que buscamos resultando perfecto para nuestra aplicación.

Existen otras tabletas del fabricante Wacom como la STU-WACOM-300 pero esta tableta dispone de una pantalla menor, concretamente de 4" lo cual resulta demasiado pequeña para firmar cómodamente y presenta una menor resolución que la escogida (396 x 100 píxeles). También podemos encontrar la tableta STU-WACOM-520 que incluye una pantalla a color de 4,7" y una resolución más alta que la escogida, pero dispone de unas características superiores de las que no vamos a hacer uso por ese motivo se descartó.

Una de las desventajas que presentan las tabletas STU-WACOM es que es necesario el lápiz que incorporan para poder realizar la firma, es decir no se puede utilizar el dedo del usuario para firmar, es necesario un dispositivo específico.

Existen otras marcas de tabletas digitalizadoras de firma en el mercado que podrían ser igual de validas que las escogidas para nuestra aplicación.

5.3.2 Teléfonos móviles con pantalla táctil

Los teléfonos móviles incorporan pantallas táctiles que podrían utilizarse para la captura digital de la firma de un usuario permitiendo desarrollar esta misma aplicación para sistemas operativos de teléfonos móviles tales como Android, iOS o similares. El problema que se presenta es que en la actualidad las pantallas táctiles de los móviles en su mayoría no son aptas para la captura de firmas digitales ya que hay parámetros que son incapaces de capturar. Muchos de los dispositivos móviles no son capaces tomar registros de presión ni de registrar la velocidad con la que el usuario realiza la firma, además la pantalla suele ser muy pequeña y resulta incómodo e impreciso para el usuario realizar la firma.

El dispositivo móvil más apto para nuestra aplicación seria el Samsung Galaxy Note II o un modelo de características similares. Este dispositivo presenta una pantalla de 5,55” además de una resolución de 1280 x 720.

Debido a la limitación de los dispositivos móviles con características apropiadas para nuestra aplicación y que existen muy pocos capaces de adaptarse a nuestras necesidades se descartaron para la realización de la aplicación basándonos en dispositivos móviles y se mantuvo la opción de utilizar las tabletas digitalizadoras de firmas.

5.3.3 Tablets

Pocas Tablets táctiles son aptas para registrar los parámetros que se requieren para caracterizar una firma digital, la mayoría no son capaces de registrar presión a la hora de realizar la firma, y por estos motivos se han descartado.

5.3.4 Otros dispositivos con pantalla táctil

Tanto los ordenadores de pantalla táctil como los terminales táctiles que encontramos en las tiendas para firma se han descartado ya que estos dispositivos en la mayoría de los casos no son capaces de registrar parámetros de la firma tan importantes como la presión o la velocidad con la que se ha firmado, tienen una mala sensibilidad y en muchos casos no resulta cómodo realizar la firma.

En definitiva los dispositivos más óptimos para realizar nuestra aplicación y que mejor se adaptan a las características requeridas son las tabletas digitalizadoras de firmas.

5.2.3 Diseño de la estructura de la aplicación

En este capítulo explicaremos el diseño escogido para la aplicación en la interacción con el usuario para acceder al sistema de recogida de muestras legítimas o al sistema de emulación de ataques de firma. A continuación mostramos la estructura general de la aplicación. Explicaremos el funcionamiento de la aplicación a través de diagramas de flujo (Ilustración 3):

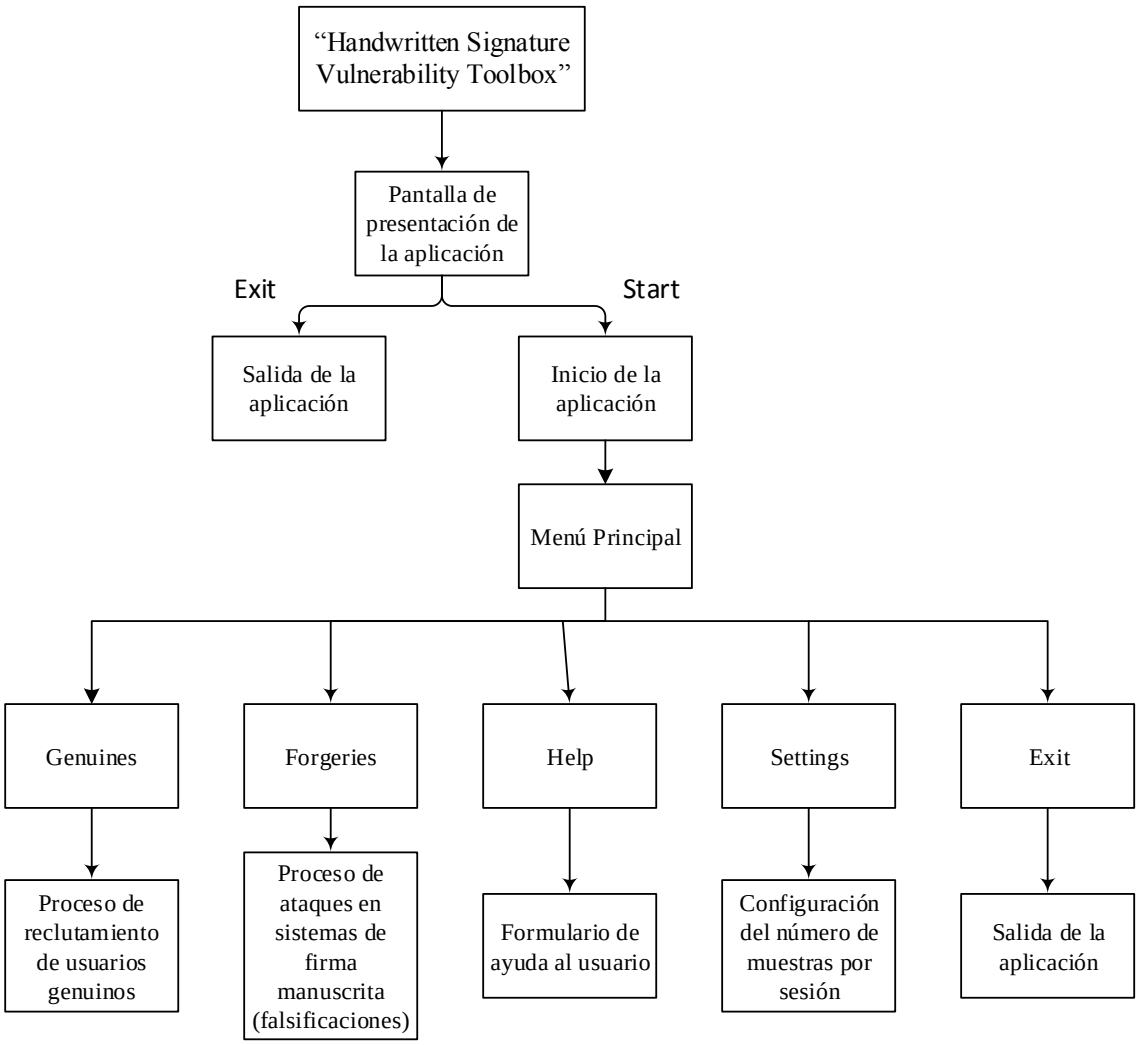

Ilustración 3 Estructura genérica de la aplicación

5.2.3.1 Forgeries

En este capítulo explicaremos el flujo de formularios que encontraremos al acceder a la aplicación "Forgeries", que es la aplicación que permite la emulación de ataques en sistemas de firma manuscrita, es decir permitirá la falsificación de firmas genuinas (Ilustración 4):

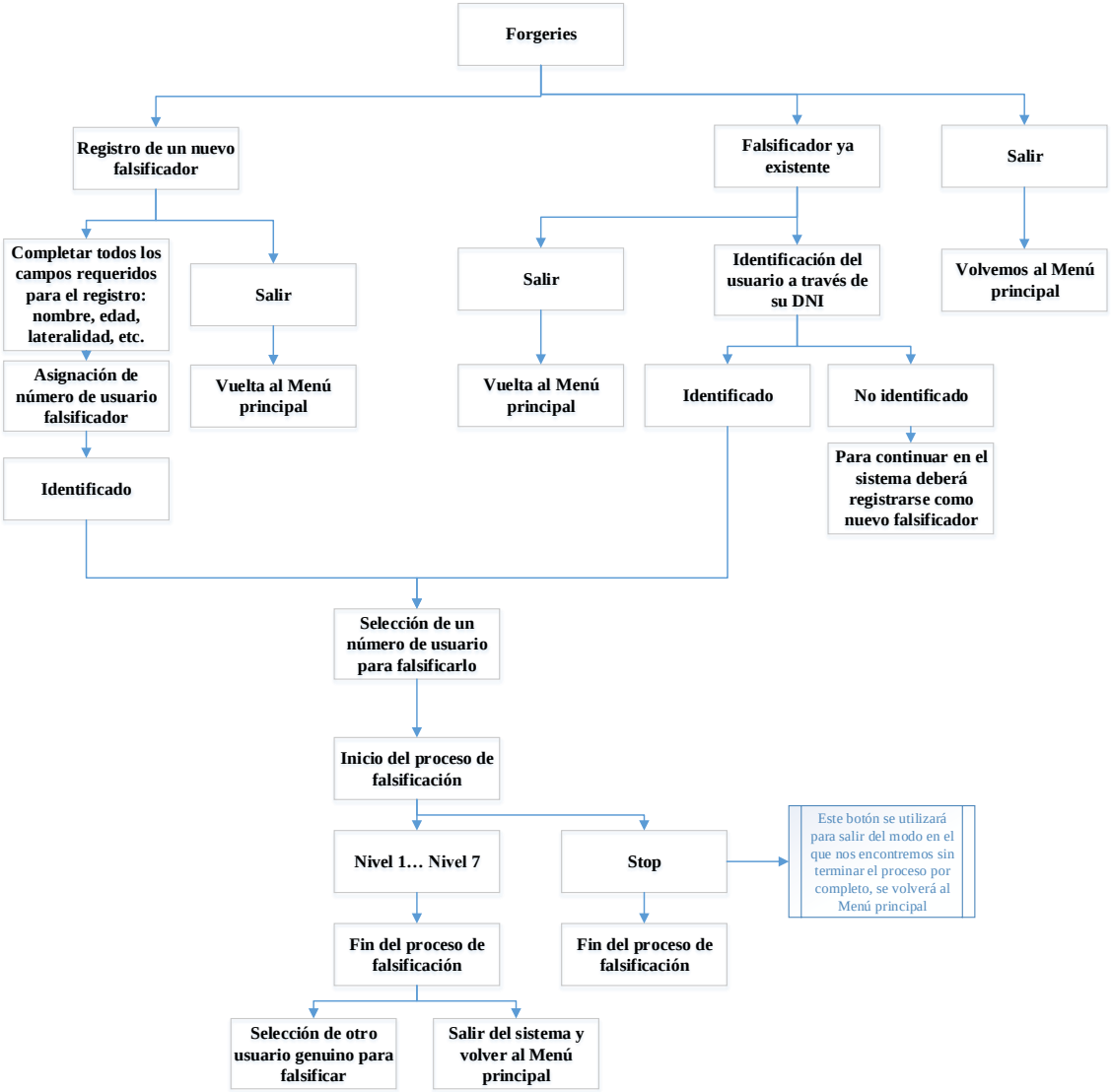

Ilustración 4 Estructura del modo Falsificador de la aplicación

Es necesario destacar que en este modo de programa puede darse el caso de que un usuario falsificador también se encuentre registrado en el sistema como usuario genuino, para este caso el usuario es registrado independientemente en las dos sub-aplicaciones, a través del DNI el sistema identifica al usuario y en el caso de estar

registrado como un usuario genuino no se le permitirá realizar la falsificación de su propia firma, es decir a la hora de seleccionar el número de usuario genuino a falsificar no aparecerá su número de usuario para falsificarlo.

También debemos destacar en este punto que una vez se ha completado el proceso de falsificación el usuario que ha sido falsificado desaparecerá de la lista de forma permanente de esa forma no se podrá falsificar más de una vez al mismo usuario ya que no tendría sentido.

5.2.3.2 Genuines

En este capítulo explicaremos el flujo de formularios que encontraremos al acceder a la aplicación "Genuines", que es la aplicación que permite la captura de firmas genuinas de los usuarios (Ilustración 5):

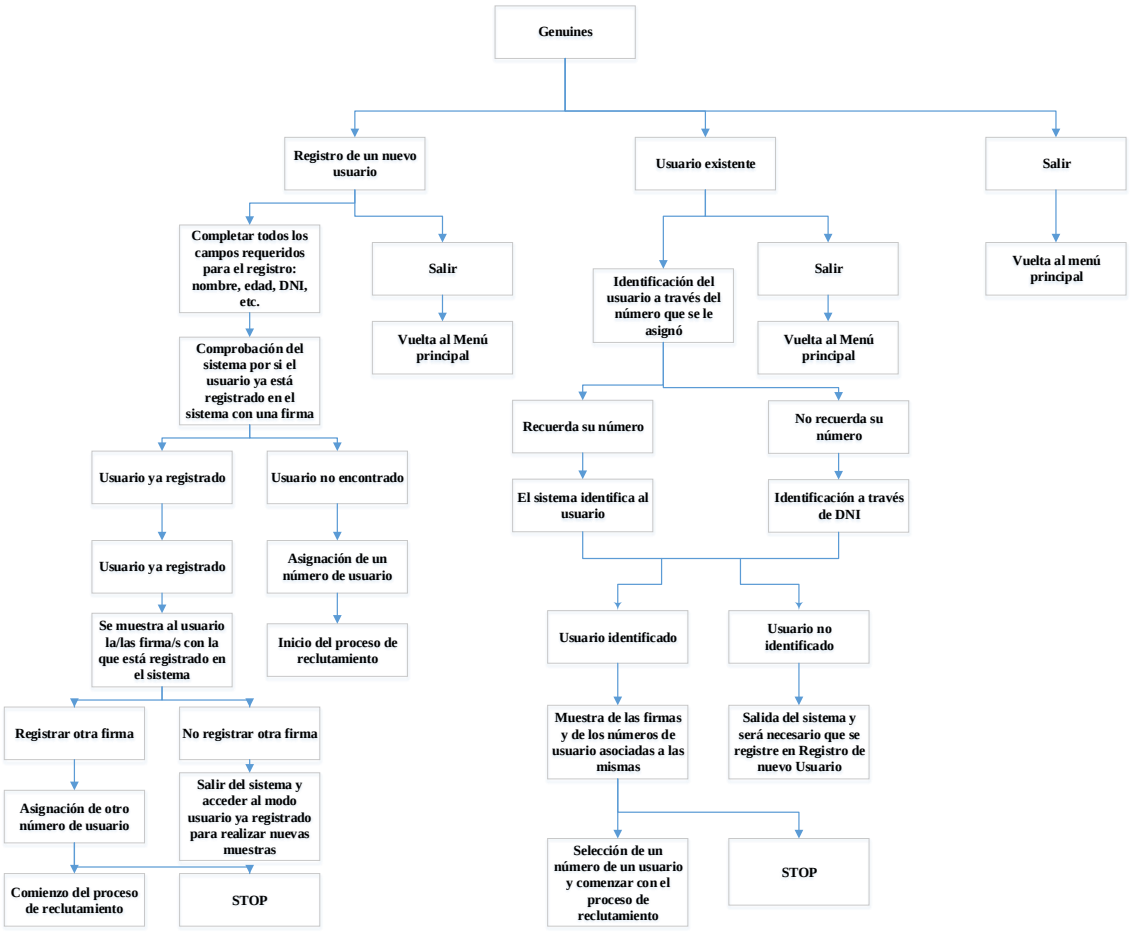

Ilustración 5 Estructura del modo de reclutamiento genuino

Destacaremos en este punto que el proceso de reclutamiento se compone de un total de 10 muestras por sesión por defecto, pero este número de muestras se puede modificar.

5.2.3.3 Help

En el menú principal de la aplicación encontramos también la opción Help (ayuda) que nos proporciona información útil sobre el funcionamiento de la aplicación, informa al usuario sobre la investigación que se está realzando y también sobre sus derechos y como van a ser utilizados sus datos y sus firmas. Es importante proporcionar información al usuario sobre estos aspectos debido a que mejoramos la interacción con la aplicación y minimizamos el posible rechazo que pueda tener el usuario respecto a dar sus datos y proporcionar sus firmas para la investigación (Ilustración 6).

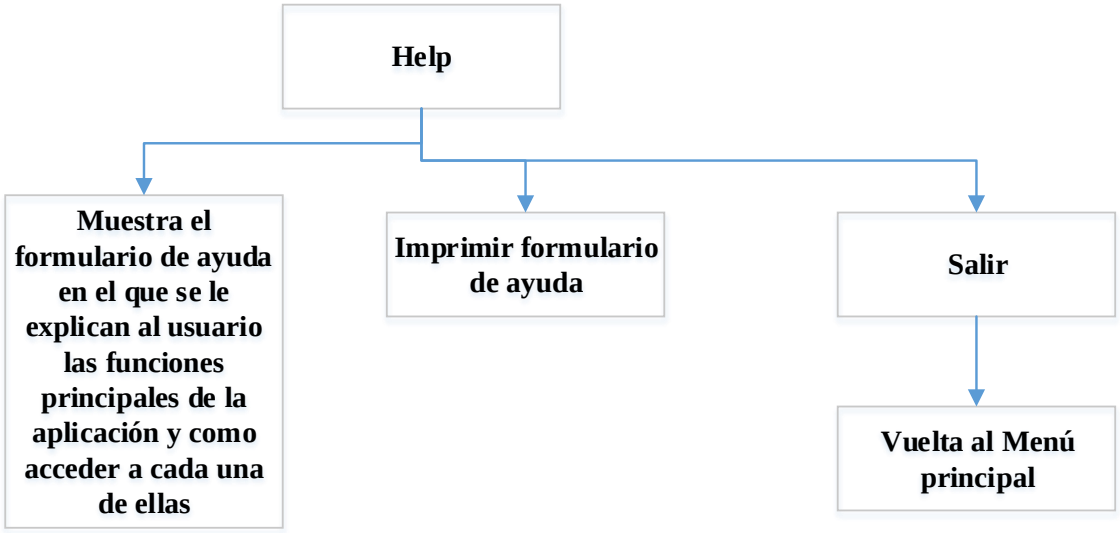

Ilustración 6 Estructura del formulario Help

5.2.3.4 Settings

Este formulario permite establecer el número de muestras a realizar por sesión en el caso de Genuines y el número de muestras por nivel en el caso de Forgeries. Por defecto están establecidas 10 muestras por sesión en Genuines y 5 muestras por nivel en Forgeries (Ilustración 7).

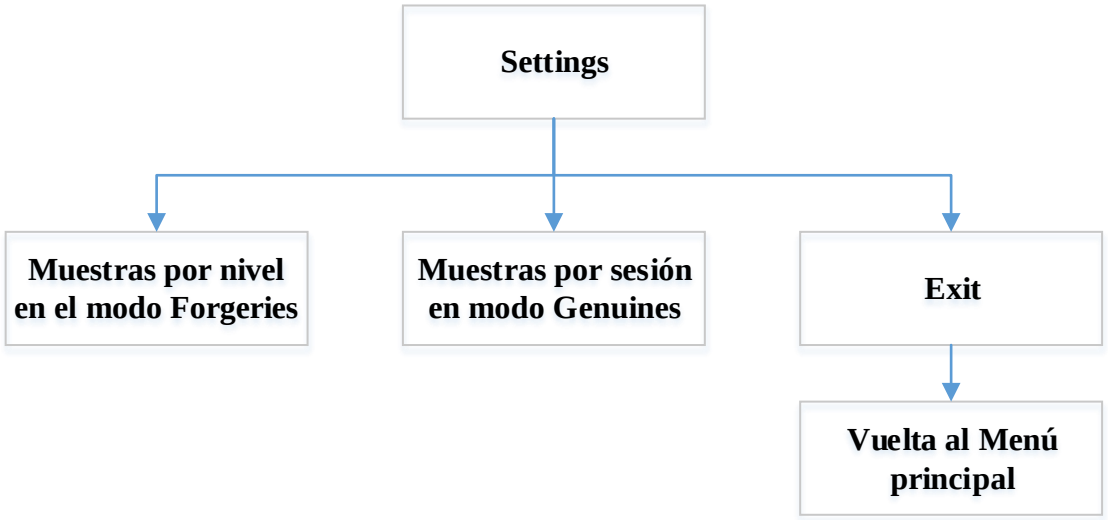

Ilustración 7 Estructura del formulario Settings

5.2.3.5 El botón STOP

El botón Stop aparece cuando estamos realizando o bien el proceso de captura de muestras de firmas genuinas o cuando estamos en el proceso de falsificación de una firma genuina, este botón permite salir del modo en el que nos encontremos y continuar en la aplicación sin haber terminado el proceso por completo.

5.2.3 Diseño gráfico de la aplicación

En este capítulo mostraremos el diseño gráfico escogido para la aplicación, la forma que presentan los formularios, y cómo se han implementado los niveles de falsificación para que al usuario le resulte cómodo realizar el proceso:

Al acceder a la aplicación "Handwritten Signature Vulnerability Toolbox" nos aparece una ventana de bienvenida permitiéndonos acceder a la aplicación o salir (Ilustración 8):

Ilustración 8. Pantalla de presentación de la aplicación

Ilustración 9. Menú

Una vez que accedemos a la aplicación tenemos un cuadro de menú en el que podemos acceder a diferentes opciones (Ilustración 9):

Para acceder al modo de la aplicación de captura de firmas genuinas tendremos que hacer click en el botón “Genuines” inmediatamente aparece un cuestionario en el que pide que nos identifiquemos como usuario existente (existing user), nuevo usuario (new user) o salir (exit) del modo registro de firma genuina, Para el caso del modo Forger el formulario que se muestra es similar (Ilustración 10).

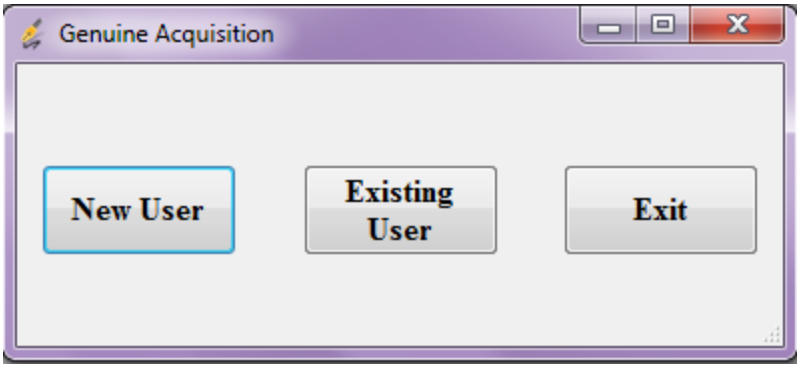

Ilustración 10. Identificación de Usuario genuino

A continuación vemos el formulario de registro de usuario genuino “Genuine”, como se ha comentado anteriormente será necesario que todos los campos estén completos para continuar el proceso, los campos del formulario de registro de usuario falsificador “Forger” son similares (Ilustración 11), simplemente encontraremos la diferencia de que no se mostrará ninguna firma ya que no es necesario:

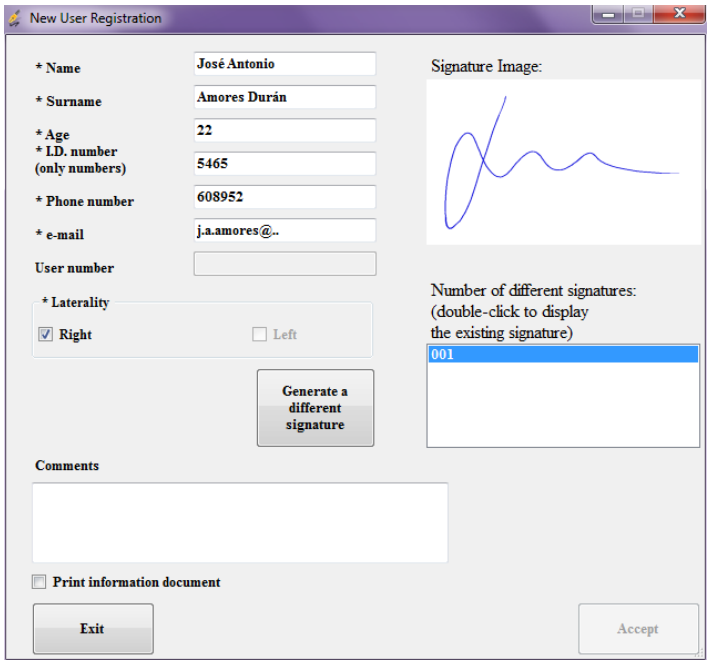

Ilustración 11. Formulario Usuario genuino

En el modo de registro de nuevo usuario existe la opción de imprimir una hoja de información en la que se explica en qué consiste el proceso de firma, para que se van a utilizar sus datos y otros tipos de datos de interés esto es válido tanto para “Genuines” como para “Forgeries”.

En la Ilustración 12 podemos observar el formulario de usuario ya registrado:

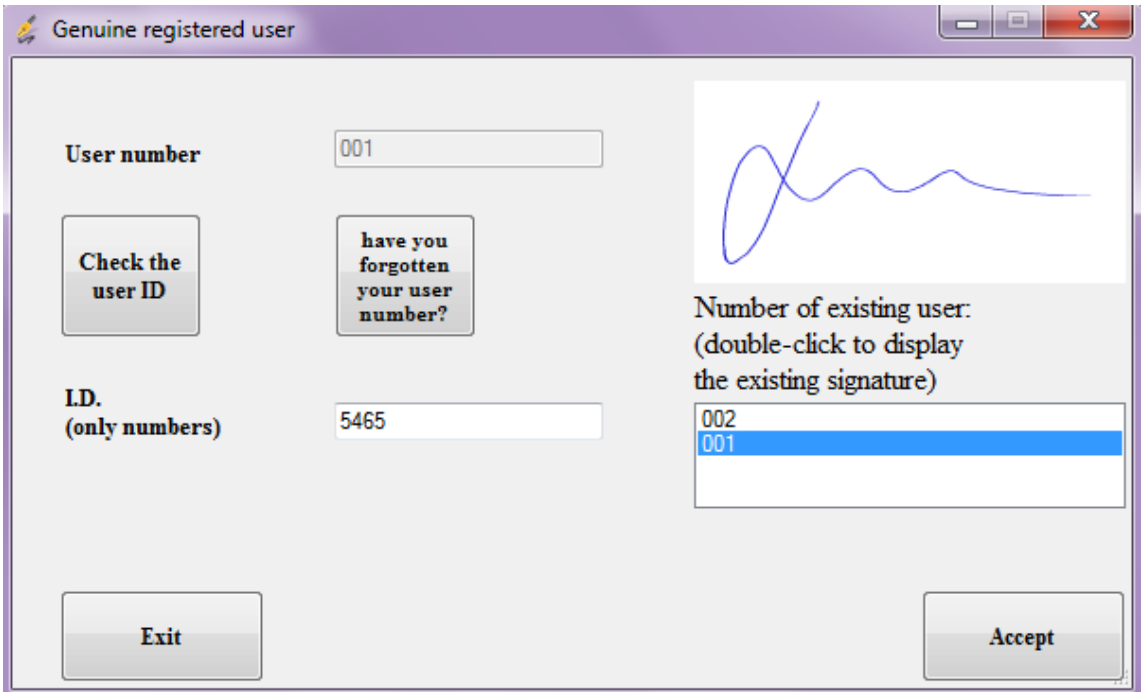

Ilustración 12. Formulario usuario genuino existente

En la Ilustración 13 podemos observar la pantalla que se le presenta al usuario en la tableta STU-WACOM para realizar las firmas:

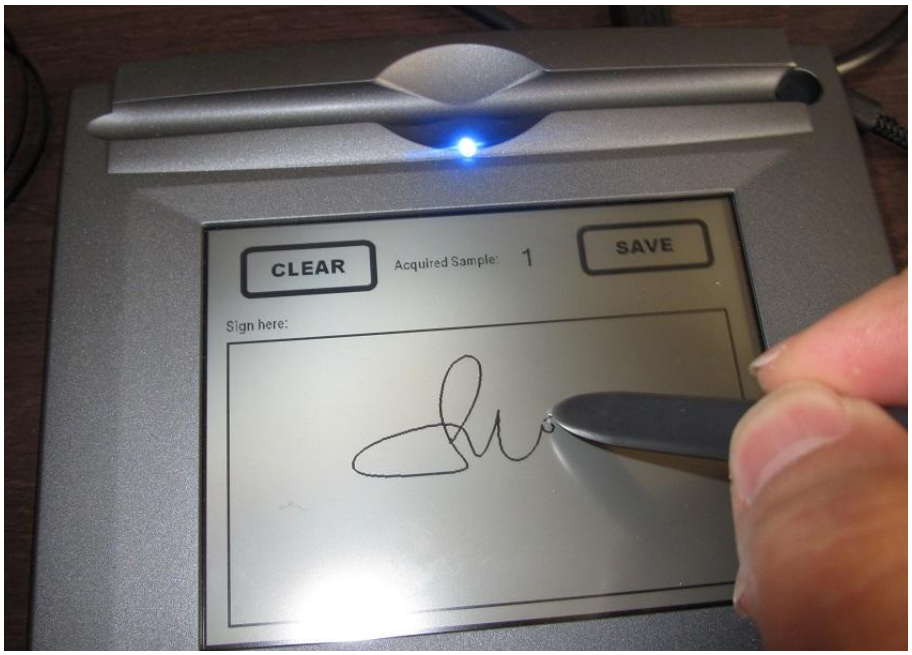

Ilustración 13. Firma en la STU-WACOM

Como podemos observar (Ilustración 13) desde la propia STU-WACOM el usuario podrá guardar la muestra si considera que ha salido correctamente o borrarla si no es correcta (pulsando “SAVE” o “CLEAR”).
La pantalla presenta un recuadro donde el usuario debe realizar la firma, también se le muestra el número de muestra que está realizando.

A continuación mostraremos el diseño gráfico de los niveles de falsificación de la aplicación, esta ha sido una de las partes más delicadas de la aplicación debido a que se ha tratado de realizar lo más intuitiva posible.

Podemos observar en la Ilustración 14 el nivel 1 de falsificación, podemos ver que no se muestra absolutamente nada ya que en este nivel no se le muestra nada al falsificador:
En la Ilustración 14se ve la lista de números de usuarios que podemos falsificar, también se mostrará por la pantalla del PC el número de muestras que llevamos.

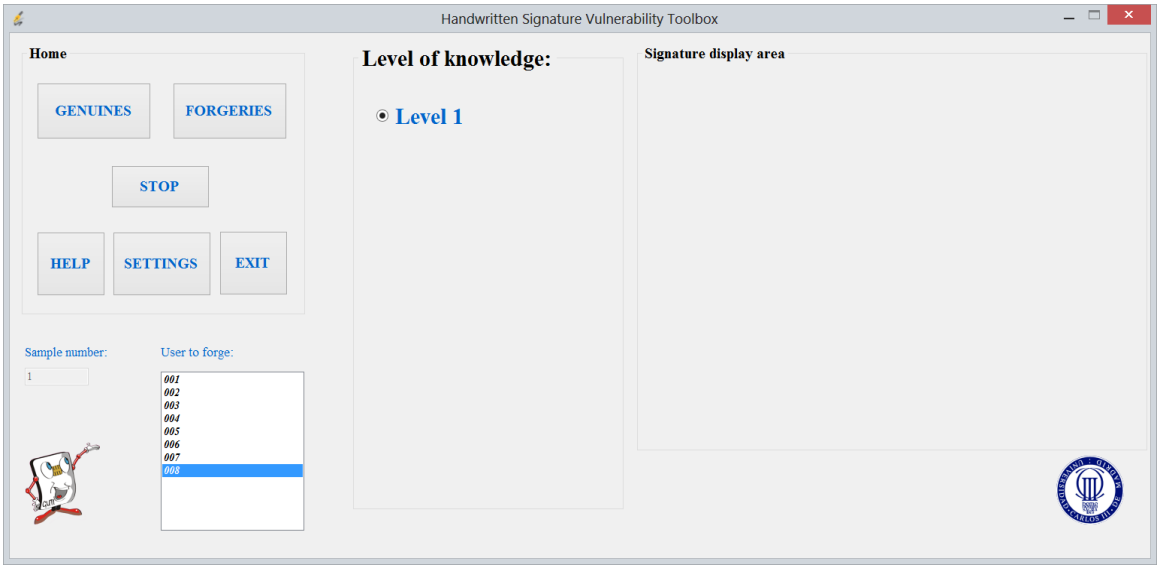

Ilustración 14. Niveles de falsificación

En la Ilustración 15 podemos observar el nivel de falsificación 3 en el que se muestra la imagen de la firma del usuario que queremos falsificar, destacamos que el diseño gráfico del nivel 2 es similar al del nivel 3:

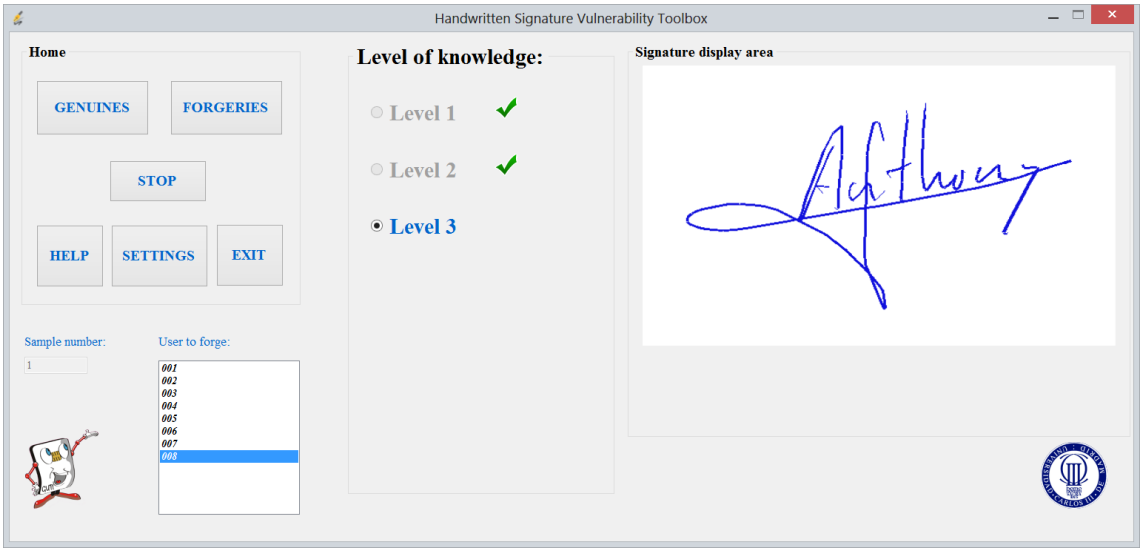

Ilustración 15. Nivel 3 de falsificación

En el nivel 4 es importante que mostremos la tableta, ya que este nivel es el de calco de la imagen de la firma que estamos falsificando (Ilustración 16):

Ilustración 16. Calcar STU-WACOM

A partir del nivel cuatro como se ha explicado en capítulos anteriores pasamos a mostrar la firma al usuario falsificador de forma dinámica, en la Ilustración 17 podemos observar el nivel 6 en el que se ve el "Reproductor de firmas" con los botones play,

pause, etc. El nivel 5 tiene un diseño similar al que se ve, tan solo encontramos la diferencia de que este nivel no contiene el "Reproductor de firmas".

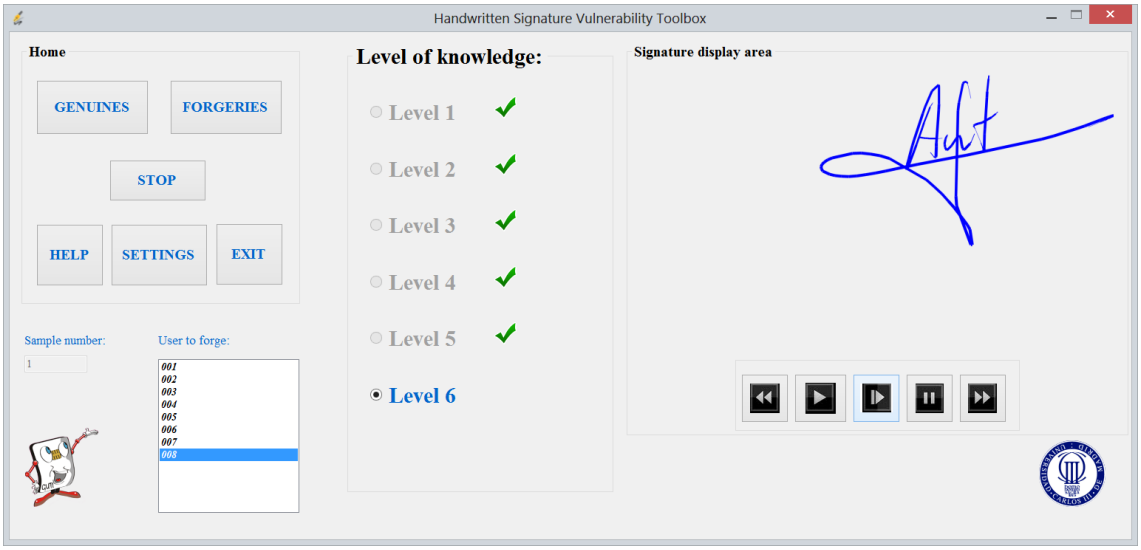

Ilustración 17. Nivel 6 de falsificación

El nivel 7 combina el diseño gráfico del nivel 6 y del nivel 4.

Una vez se hayan completado todos los niveles se le mostrará al usuario un mensaje informándole de que el proceso se ha realizado de forma correcta. El usuario que el falsificador acaba de falsificar desaparecerá de la lista (puesto que ya no tendría sentido que volviera a realizar las falsificaciones ya que conoce todas las características de la firma), y podrá realizar el proceso con otro usuario si lo desea, en caso contrario puede salir de este modo.

En la Ilustración 18 se puede observar el diseño gráfico escogido para el formulario de ayuda:

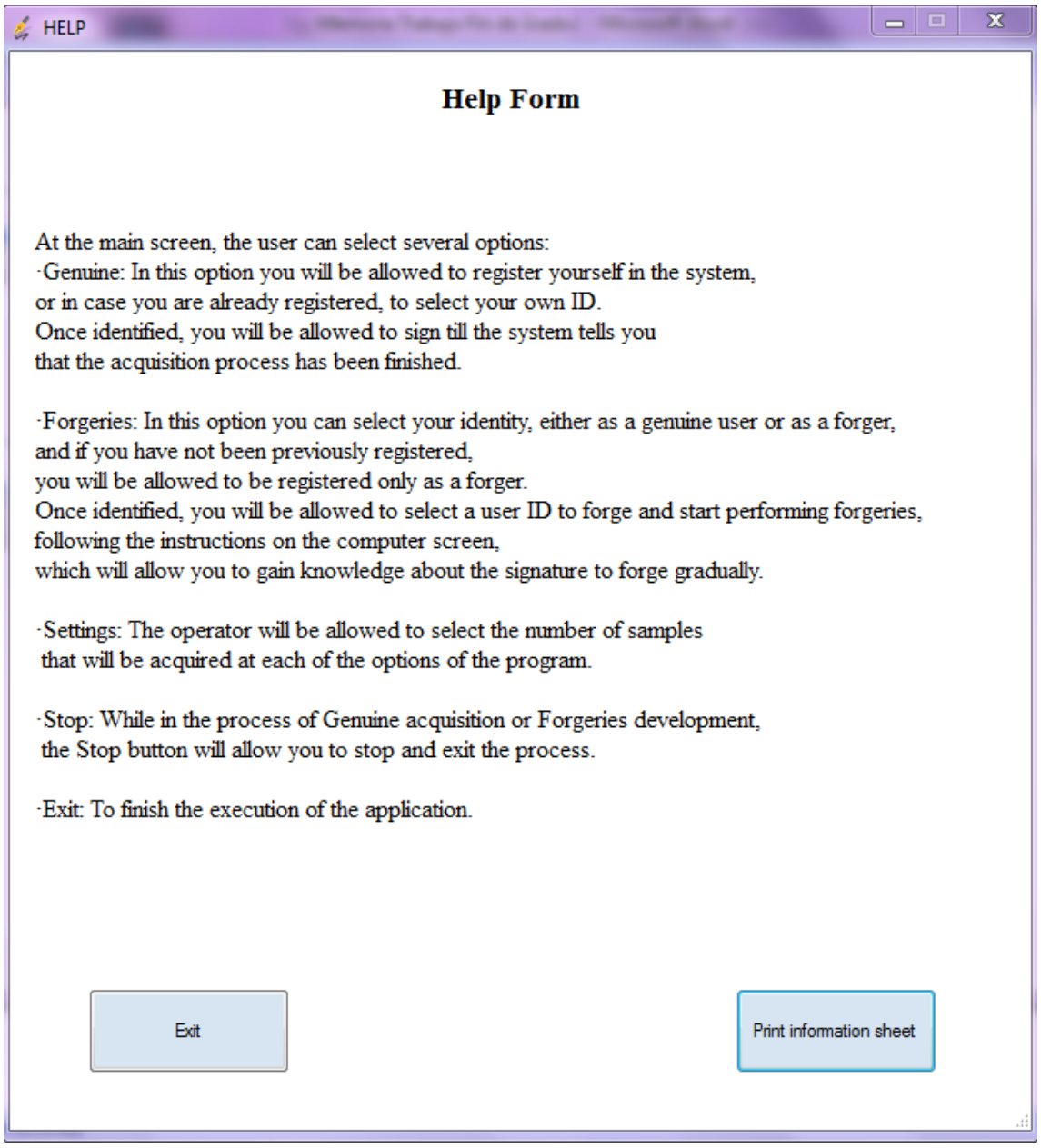

Ilustración 18. Formulario Ayuda

6 Desarrollo de la aplicación

En este capítulo se definirá la forma en la que hemos realizado la aplicación, es decir, explicaremos como hemos llevado a cabo el diseño descrito en el capítulo anterior. Se explicarán los problemas que han ido surgiendo en el desarrollo de la aplicación y cómo se han solucionado.

6.1 Inicio del desarrollo de la aplicación

En un principio fue necesario realizar un estudio sobre técnicas biométricas basadas en firmas manuscrita, por lo que comenzamos buscando documentación sobre esta técnica biométrica. Fue necesario realizar un estudio de las dos filosofías de firma manuscrita: La firma dinámica y la firma estática y buscar la forma de integrarlas en la aplicación.

Posteriormente fue necesario estudiar la tableta digitalizadora de firmas para ver su comportamiento y entender su funcionamiento, esto fue básico para su integración en la aplicación.

También fue necesario un estudio sobre el entorno y el lenguaje de programación utilizado por lo que se tomó la documentación de la página oficial de Microsoft MSDN: http://msdn.microsoft.com/en-US/. Se comenzó realizando un primer proyecto desde el principio en Visual Studio y creando una aplicación de Windows Forms simplemente para entrar en contacto con el entorno y la programación que se ha utilizado para desarrollar el proyecto.

6.2. Librerías adicionales empleadas en el desarrollo de la aplicación.

Debido a que necesitamos la interacción entre la tableta STU-WACOM y la aplicación fue necesario incluir unas referencias que nos proporcionó el fabricante.
Es necesario que nuestra aplicación llame a diversos métodos que se incluyen en los archivos.dll que provienen de otro ensamblado. Concretamente fue necesario incluir las siguientes librerías para que nuestra aplicación funcionase correctamente:
FullFormatSignature19794, libeay32, msvcr71, SignCaptureWacomSTU, STUTablet, STUTabletCore, winusb y zlib1.

A continuación explicaremos las más importantes:

6.2.1 FullFormatSignature19794

Al incluir esta librería [16] .dll en nuestro programa conseguimos guardar las firmas en un formato correcto, también conseguiremos leer los archivos que utilizaremos en la

aplicación (archivos que contendrán los datos de las firmas). Las firmas se guardarán en archivos.bir y estos archivos se guardarán y leerán mediante esta librería.
Un archivo de firma en formato completo estará compuesto por los siguientes elementos:

- Un encabezado general que contiene la información descriptiva sobre la estructura y el contenido de registro de datos.
- Un "Record Body" que contiene al menos una representación de la firma.

Todo esto se incluirá en una única clase que se denominará "Record" que a su vez se compone de otras clases como se detalla en el siguiente esquema, todas estas clases se podrán utilizar posteriormente para guardar y visualizar las firmas que capturemos con la tableta.

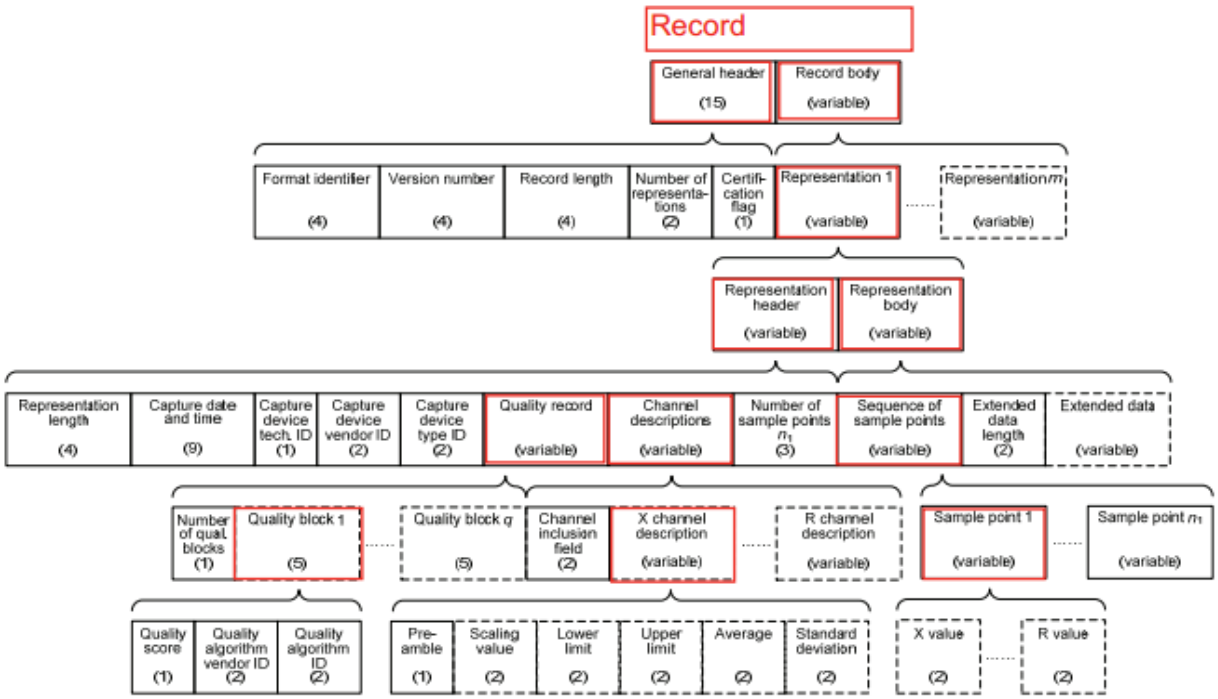

Ilustración 19. Full Format Signature [16]

Como podemos comprobar (Ilustración 19) tenemos una clase genérica Record que se compone de diferentes propiedades y métodos con los que conseguiremos leer los archivos de firma y guardarlos en un correcto formato.

Dentro de la librería FullFormatSignature19794 encontramos la clase "Record" que engloba las demás clases.

Cada una de estas clases guarda un tipo de dato relacionado con la firma y la tableta, de forma que conseguimos todos los datos necesarios de los que se compone la firma.

Así la clase Representation Body es la que nos proporciona la información sobre las posiciones (X, Y) de la firma, su presión y el tiempo que se ha empleado en firmar.

Tenemos que destacar en este capítulo que la librería FullFormatSignature19794.dll se basa en la normativa ISO/IEC 19794-7. Por lo que es importante entender esta normativa para poder comprender el funcionamiento de esta librería.

6.2.1.1. Normativa ISO/IEC 19794-7

Todas las muestras de firmas que tomamos en la aplicación se basan en esta normativa por lo que explicaremos a continuación en qué consiste:

Esta normativa tiene como objetivos especificar los formatos de almacenamiento de datos para las firmas en forma de series temporales utilizando dispositivos tales como tabletas digitalizadoras (en nuestro caso STU-WACOM) para conseguir interoperabilidad entre los diferentes sistemas biométricos avanzados y las aplicaciones. En ella se define la estructura de datos para almacenar las firmas y una descripción de su contenido.

Los datos capturados se almacenan por canales, que almacenan la información recogida, es decir, los diferentes tipos de información, tales como la posición del bolígrafo, velocidad de escritura, etc. La denominación de dichos canales es en función de la información que contienen y son los siguientes:

- Canales de posición: a la hora de digitalizar los datos, se define un sistema de coordenadas tridimensional, tres canales graban la posición tridimensional del bolígrafo. En este caso la unidad de medida es el metro.

    - X: coordenada X o posición horizontal del bolígrafo: incrementa su valor de coordenada hacia la derecha.

- Y: coordenada Y o posición vertical del bolígrafo: el incremento de su valor es hacia arriba.

- Z: coordenada Z o altura del bolígrafo sobre el plano de escritura: movimiento del bolígrafo levantando o presionado la tableta, obtiene su valor de coordenada según aumenta la presión.

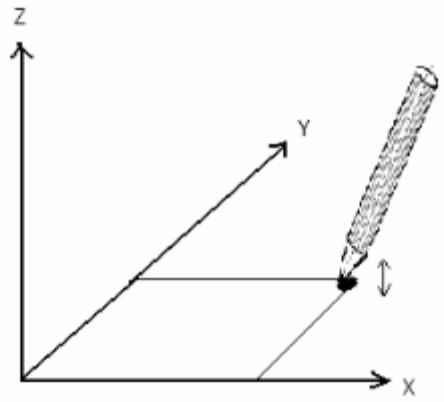

Ilustración 20. Canales de posición. Eje de coordenadas

Podemos ver (Ilustración 20. Canales de posición. Eje de coordenadas) los canales de posición, es decir el eje de coordenadas.

- Canales de velocidad: indica la rapidez con la cual se escribe la firma. La unidad de medida es el m/s.

  - VX: velocidad en el plano horizontal.
  - VY: velocidad en el plano vertical.

- Canales de aceleración: almacena la diferencia de velocidades, es decir, la variación de velocidad de escritura a lo largo de toda la firma. La unidad de medida es el m/s^2.

  - AX: aceleración en el plano horizontal.
  - AY: aceleración en el plano vertical.

- Tiempo: tiempo transcurrido desde el comienzo de la toma de datos. La unidad de medida es el segundo.

- Diferencia temporal: diferencia temporal de la muestra anterior con respecto a la actual.

- Presión: valor asociado a la presión con la que se actúa con el bolígrafo sobre la tableta. La unidad de medida es el Newton.

- Estado: almacena “0” si el bolígrafo no está tocando la tableta y “1” si por el contrario hay contacto.

- Orientación: mide la inclinación del bolígrafo con respecto a los ejes. La unidad de medida es el ángulo (º).

El formato que explicamos en nuestra librería es el formato completo.

6.2.2. SignCaptureWacomSTU

Mediante esta librería [17 se creó el control SignControlWacomSTU [18] que es el control que utilizamos para capturar firmas en nuestra aplicación, este control es capaz de controlar eventos tales como el comienzo de la realización de la firma, si la tableta está conectada o no está conectada al ordenador, además incluye unos controles que nos sirven para desde la tableta aceptar (pudiendo guardar la firma) o cancelar (que nos permite borrar la firma realizada).

De la misma manera en este control se definirán los botones de la tableta:

- Botón aceptar: que se define mediante las coordenadas X e Y de la STU-WACOM. Es decir, para definir en la pantalla de la tableta los controles X e Y definiremos mediante dos posiciones X (X1, X2) las posiciones del control en el eje horizontal, y definiremos mediante dos posiciones Y (Y1, Y2) las posiciones verticales de nuestro botón en la tableta. Una vez definidos procederemos a configurarlo de tal modo que cuando el usuario presione sobre la tableta los el control definido salte un evento que en este caso será el de guardar firma.

- Botón cancelar: que se define de la misma manera que el botón anterior con la única diferencia de que cuando lo presionamos el evento que saltará será el de borrar la pantalla, eliminando la firma que había sobre ella (el botón de cancelar la pantalla es el que se encarga de borrar la firma, este evento viene directamente llamando a la función signControlWacomSTU1.cleanSignature();).

A través de este control procedemos a guardar los datos de la firma que se realiza en la pantalla de la tableta. También se mostrará dicha firma en la pantalla del control.

El procedimiento para guardar los datos que se generan en la tableta es mediante la clase ArrayList cuyos objetos utilizan una matriz cuyo tamaño aumenta dinámicamente

cuando es necesario, de esta forma se podrán guardar tantos puntos como sean necesarios para definir una firma.

Utilizaremos objetos tipo ArrayList para guardar las posiciones X e Y que definan la firma, así como la presión y el tiempo. Con todos estos elementos seremos capaces finalmente de crear un archivo de extensión .bir que contenga los principales parámetros de una firma. En la ilustración 31 se muestra el control descrito.

Ilustración 21. SignControlWacomSTU

Una vez han sido implementadas estas librerías en nuestro proyecto podemos comenzar a definir nuestras aplicaciones.

6.3. Desarrollo de la aplicación para captura de firmas genuinas

Esta fue la primera aplicación que se definió para la aplicación "Handwritten Signature Vulnerabilities Toolbox", ya que en primer lugar necesitábamos tener firmas genuinas para conseguir implementar la aplicación de emulación de ataques de firma manuscrita.

Se comenzó como hemos explicado añadiendo las librerías anteriormente descritas en nuestro proyecto y se agregó en el formulario de proyecto Form1 el control SignControlWacomSTU [18] con el que podemos guardar los datos de la firma que queremos capturar y también controlar distintos eventos como la desconexión de la tableta.

6.3.1. Desarrollo del formulario de registro de usuario genuino

Al pulsar el botón de registro de nuevo usuario del formulario Form5. "New User" tenemos que dar acceso a otro formulario nuevo en el que podamos tomar los datos del usuario que estamos registrando por lo que desde el formulario principal se activará un

objeto de tipo booleano (solo puede ser true o false), es decir este objeto pasará a ser true y por consiguiente aparecerá el formulario de registro de usuario genuino o como lo hemos llamado por código Form2.

En este formulario será necesario que el usuario complete una serie de campos por lo que utilizaremos controles de tipo TextBox que son controles específicos para rellenar campos por usuarios. Concretamente hemos necesitado de 8 TextBox para completar todos los campos que se requieren (nombre de usuario, edad, etc.).

Todos estos datos serán guardados en un archivo .txt en la misma ubicación donde se guardarán los datos del usuario que estamos registrando. Para ello necesitamos crearnos un objeto de la clase StreamWriter que se encargará de recoger los datos que se encuentran en los campos de cada uno de los TextBox y escribirlos en un archivo .txt que posteriormente guardará.

Necesitamos controlar todos los TextBox ya que es necesario que todos estén completos antes de continuar con el proceso y poder generar un número de usuario, por lo que necesitaremos controlar si todos los campos han sido cubiertos con texto, para el reconocimiento de si el usuario es diestro o zurdo hemos utilizado objetos del tipo CheckBox ya que se considera más cómodo para que el usuario opere con el sistema, de la misma manera es necesario que el usuario seleccione una de las dos opciones del CheckBox para continuar (diestro o zurdo).

Para la asignación del número de usuario tenemos que comprobar los archivos que existen en el directorio donde estamos guardando nuestras firmas genuinas, si no existe ninguno crearemos un nuevo directorio con el nombre “001” que corresponderá con el número de usuario que se acaba de registrar y dentro de esa carpeta se guardarán todos los archivos (.txt, .bir y .bmp) correspondientes a dicho usuario.

En este formulario se contempla la posibilidad de que el usuario genuino exista en el sistema con otro número de usuario, es decir, esté registrado con otra firma, para este caso tenemos que comprobar su número de DNI.

La aplicación guarda todos los números de DNI de los usuarios que se registran asignados a su número de usuario de tal forma que para casos como el que se presenta podemos encontrar rápidamente si el usuario existe en el sistema con otro número de usuario (y por lo tanto con otra firma) o no existe. Lo que hacemos para comprobar su existencia es comparar los DNI del archivo que tenemos guardado con el DNI que el usuario acaba de introducir en el sistema, en el caso de que coincida guardamos el numero de usuario en un ArrayList y activamos un objeto de tipo booleano como encontrado=true;. Esto lo conseguimos creándonos un objeto capaz de leer ficheros de texto que tendrá que pertenecer a la clase StreamReader. Finalmente y una vez realizada la comprobación pondremos todos los números de los usuarios encontrados en una

ListBox, que es un control donde el usuario puede seleccionar los elementos que aparecen en él, donde le daremos la opción al usuario de poder ver una vez seleccione su número correspondiente la firma que realizó en la sesión previa y podrá decidir si hacer otra firma o salir del modo de registro o entrar en el modo de usuario registrado.

Para la visualización de la firma tendremos que una vez el usuario ha seleccionado el número de usuario que se corresponde con el de la firma que tenemos que buscar el sistema buscará en el directorio de firmas genuinas con el mismo número que el número de usuario seleccionado, entrará en la carpeta y seleccionará el primer archivo .bmp correspondiente a la firma de dicho usuario. Una vez seleccionada dicha firma se expondrá en un PictureBox que es un componente específico para la mostrar imágenes. Y de esta forma el usuario podrá ver la firma que realizó en su momento.

En el caso de que la búsqueda no encuentre coincidencias con el DNI del formulario el booleano encontrado=false; y se le asignará directamente un nuevo número de usuario, desbloqueándose el botón del formulario "Accept" y permitiendo continuar el proceso.

En este formulario existe también la posibilidad de imprimir la hoja de registro del usuario, simplemente antes de pulsar "Accept" tenemos que seleccionar el TextBox correspondiente, si el TextBox es seleccionado se activará el proceso de imprimir al pulsar "Accept".

6.3.2. Desarrollo del formulario de usuario genuino existente

Si por el contrario pulsamos la opción de usuario genuino ya registrado "Existing User" tenemos que crear un nuevo formulario al que llamaremos Form3.

En este formulario tendremos que introducir para identificar al usuario en el sistema su número de usuario. Una vez introducido el número del usuario que queremos identificar el sistema buscará en el archivo .txt donde hemos registrado todos los números de usuarios con sus respectivos DNI mediante un objeto de la clase StreamReader y compararemos el numero de usuario introducido con los datos de la lectura que ha recogido el objeto StreamReader. Si coincide le indicaremos al usuario que ha sido identificado, en caso contrario el usuario deberá salir de este modo y registrarse como nuevo usuario.

Puede darse el caso de que el usuario no recuerde el número que se le asignó, por lo que fue necesario incluir una alternativa de identificación. Se le permite al usuario introducir su DNI en el caso de no recordar su número de usuario, de esta forma se buscará en el archivo .txt si el número de DNI coincide, esta búsqueda se hará de la misma forma que anteriormente, mediante un objeto de la clase StreamReader. En el caso de que la búsqueda encuentre al usuario hay que tener en cuenta que este puede tener más de una firma, se mostrarán en un ListBox todos los números de usuario relacionados con el

número del DNI introducido de tal forma que el usuario podrá seleccionar el que desee, una vez seleccione un numero de usuario de la lista se le mostrará la imagen de la firma que tiene asignada, permitiéndole al usuario recordar con que firma realizó el registro.
Una vez identificado completamente se podrá continuar con el proceso pulsando "Accept".

Hay que destacar que para tanto el sistema de registro de nuevo usuario como para el caso de usuario registrado el botón "Accept" permanecerá desactivado hasta que no hayamos completado todos los campos requeridos y nos hayamos identificado correctamente, de esta forma evitamos errores en el sistema. Este control se lleva a cabo mediante el control de eventos de cada uno de los controles del formulario, se ha explicado un ejemplo anteriormente.

También es común la forma en la que se ha programado el botón "Exit" de los dos formularios de identificación, es decir, este botón únicamente se utilizará para salir del formulario en el que nos encontramos. Se programará Form.Close();

Finalmente una vez los usuarios se han identificado como usuarios genuinos nuevos o existentes en el Form1 (formulario principal) se activará un booleano con el que controlaremos la forma en la que se guardan los datos, si el usuario ha realizado el registro de nuevo usuario se activará el booleano correspondiente que permitirá acceder dentro del evento Click del botón "Save" a la forma de guardar propia de este caso.

Conseguimos controlar esto mediante un bucle "if" (si el booleano es true accedemos al código que engloba el bucle), la mayoría de las condiciones se han realizado mediante este bucle.

Se hará de la misma manera para el usuario registrado.

6.4. Desarrollo de la aplicación de emulación de ataques de firma manuscrita

Una vez que hemos conseguido realizar correctamente la aplicación para reclutar las firmas genuinas podemos realizar la aplicación de emulación de ataques de firma manuscrita que es la más compleja y donde más problemas han surgido.

Para esta aplicación nos sirven las mismas librerías que utilizamos anteriormente, por lo que utilizaremos el mismo control para la captura de las firmas (el SignControlWacomSTU[18]).

Comenzaremos explicando el registro e identificación de los usuarios falsificadores en el sistema, el procedimiento a seguir ha sido muy parecido al que hemos utilizado en la aplicación de captura de firmas legítimas.

Se ha incluido en el formulario principal otro botón llamado “Forgeries” que será el que el usuario deberá pulsar para acceder a la aplicación. El evento de este control lleva asignado que cuando se hace click sobre él se activará un booleano que permitirá acceder al código propio de esta aplicación, a su vez desactivará el booleano que permite que se active el código perteneciente a la aplicación de reclutamiento de firmas genuinas, de esta forma solucionamos el problema de tener dos sub-aplicaciones implementadas en una. Se ha aplicado lo mismo para el caso de usuarios genuinos.

Cuando el usuario pulsa sobre el control “Forgeries” aparecerá un nuevo formulario de registro en el que se podrá seleccionar un nuevo usuario falsificador, un falsificador ya registrado o salir del formulario y volver al formulario inicial, este formulario lo hemos llamado Form10.

6.4.1. Desarrollo del formulario de registro de nuevo falsificador

Para este caso tenemos que al seleccionar nuevo falsificador, si seleccionamos nuevo falsificador aparecerá el formulario de registro del falsificador, que en este caso será el Form9.
Este formulario es más sencillo de implementar que el de registro de usuarios genuinos debido a que no se dará el caso de que exista un falsificador con “varias firmas”, simplemente pediremos que se completen los campos como nombre, edad, número de teléfono, e-mail, etc. Al igual que en el formulario de la aplicación de adquisición de firmas genuinas no se permitirá continuar en el proceso hasta que no se completen todos los campos requeridos.

También tenemos que asignar un número de usuario en el sistema al falsificador y para ello lo que hacemos es al igual que en el formulario de usuarios genuinos comprobamos los directorios existentes en la carpeta Firmas Falsas que creamos en los recursos del programa y comprobamos las carpetas existentes, en el caso de que no exista la carpeta existente crearemos una nueva y la numeración correspondiente a esta nueva carpeta será el número del usuario.

Destacamos en este punto el problema de no poder permitir que un usuario falsificador falsificara su propia firma legítima en el caso de estar registrado como usuario genuinio, para ello llevamos a cabo la siguiente solución:

Si el usuario que se está registrando está registrado en el sistema como usuario genuino, es decir, también ha realizado el proceso de reclutamiento de firmas genuinas y ha realizado su firma legítima, para este caso la búsqueda se realiza por el DNI del usuario, compararemos el DNI introducido con los DNI correspondientes a los usuarios genuinos registrados, cuando detectemos un DNI coincidente guardaremos en un ArrayList los números de los usuarios a los que están ligado y no mostraremos estos

números de usuario para que el falsificador los “falsifique” de esta forma solucionamos el problema. La forma de leer los archivos de texto .txt la realizamos con un objeto de la clase StreamReader y compararemos los datos mediante herramientas de código, posteriormente se almacenan los datos deseados en un ArrayList.

6.4.2. Desarrollo del formulario de usuario falsificador existente

Para este caso tenemos que el formulario correspondiente es el Form11. En este caso al igual que anteriormente tenemos que identificar si el usuario que se está autentificando está registrado en el sistema como usuario genuino, es decir, también ha realizado el proceso de reclutamiento de firmas genuinas y ha realizado su firma legítima, para este caso la búsqueda se realiza por el DNI del usuario, compararemos el DNI introducido con los DNI correspondientes a los usuarios genuinos registrados, cuando detectemos un DNI coincidente guardaremos en un ArrayList los números de los usuarios a los que están ligado y no mostraremos estos números de usuario para que el falsificador los “falsifique” de esta forma solucionamos el problema. La forma de leer los archivos de texto .txt la realizamos con un objeto de la clase StreamReader y compararemos los datos mediante bucles del tipo if y for, posteriormente se almacenan los datos deseados en un ArrayList.

Por este motivo en este formulario únicamente se permite que el usuario introduzca su número de DNI no el número de usuario ya que con el número de usuario no somos capaces de encontrarle en el sistema como usuario legítimo.

También al registrar los usuarios falsificadores llevamos a cabo un registro de sus DNI asignados a su número de usuario, estos datos estarán guardados en un archivo .txt que tendremos que leer mediante un objeto de la clase StreamReader para localizar sus datos. El proceso a seguir es el mismo que el que se ha presentado en los casos anteriores.

Cuando se localiza el usuario por su DNI se mostrará el numero de usuario que tiene asignado y podrá continuar en el proceso, en el caso de no existir deberá salir del sistema y registrarse correctamente.

6.4.3. Desarrollo de los niveles de falsificación

Una vez los usuarios falsificadores se han registrado en el sistema se podrá continuar con el proceso de falsificación de las firmas genuinas de tal forma que se mostrará en un ListBox los números de usuarios genuinos a los que puede falsificar.
Una vez el falsificador seleccione un número de usuario genuino para falsificar podrá comenzar a realizar las falsificaciones con ayuda del nivel 1.

Todo el apartado de los niveles de falsificación se realizará en el formulario principal Form1.

Los niveles de falsificación se han realizado mediante un control RadioButtom el cual el usuario deberá seleccionar con el ratón para comenzar en ese nivel.

Hemos utilizado las propiedades Visible y Enabled de los controles para evitar que el usuario pueda saltarse niveles o repetirlos de la siguiente manera: únicamente el usuario falsificador podrá ver y por lo tanto acceder a aquellos controles RadioButtom que le corresponda, es decir, primero se le mostrará el RadioButtom correspondiente al nivel 1, posteriormente el nivel 2, y así sucesivamente hasta el nivel 7. Además no se le permitirá repetir niveles debido a que cuando finalice la realización de las muestras con uno éste pasará a inhabilitarse (Enabled=false;) y no podrá volver a acceder a él.

- Nivel 1: en este nivel al falsificador no se le mostrará nada de la firma que va a falsificar, es una versión ciega, por lo tanto lo único que hace falta es que el usuario realice las muestras oportunas, el procedimiento a seguir es el mismo que con las firmas genuinas se guardarán los dos archivos correspondientes al usuario falsificador (.bir y .bmp) con los siguientes datos:

    - El número del usuario genuino al que ha falsificado.
    - El número de usuario falsificador que ha realizado la muestra.
    - El número de muestra falsificada.
    - El nivel utilizado de falsificación.

  La forma en la que guardamos los archivos es equivalente para todos los niveles.

- Nivel 2: en este nivel se le permite al usuario visualizar durante un breve periodo de tiempo (se ha configurado 5s) la imagen de la firma del usuario que va a falsificar. Para realizar este nivel hemos tenido que utilizar un control temporizador para que transcurrido un tiempo se elimine la imagen de la pantalla.

  Una vez que el usuario falsificador ha seleccionado del ListBox el usuario que quiere falsificar el sistema buscará en el directorio correspondiente a ese número de usuario los archivos que definen su firma (.bir y .bmp, se ha escogido la primera muestra) y para este nivel únicamente necesitamos el archivo .bmp que se mostrará en la pantalla del Form1 en un PictureBox.

  Para controlar el tiempo hemos tenido que utilizar un objeto de la clase Timer. Los objetos de la clase Timer permiten especificar un intervalo recurrente en el que se provocará un evento, en nuestro caso especificamos un tiempo de 5s y

cuando se cumpla la imagen que aparece en el PictureBox será sustituida por un Bitmap en blanco de esta manera solucionamos el problema.

- Nivel 3: en este nivel tenemos que mostrar la imagen de la firma del usuario genuino de forma permanente por lo que el procedimiento para el desarrollo de este nivel es el mismo que para el nivel 2, solo que en este caso no necesitaremos un objeto de la clase Timer, ya que no es necesario controlar ningún evento. Debido a que la imagen al finalizar la toma de muestras tiene que desaparecer cambiaremos la imagen de la firma del usuario por un Bitmap en blanco.

- Nivel 4: este nivel permite al usuario "calcar" la imagen de la firma del usuario que está falsificando, en este nivel el usuario únicamente interactuará con la tableta STU-WACOM, por lo que tendremos que mostrar la imagen en la pantalla del dispositivo. En este nivel surgieron algunos problemas que a continuación contamos como se solucionaron:

  En primer lugar al introducir una imagen en la tableta fue necesario modificar la pantalla de firma de la tableta así como el tamaño de los botones de la misma para guardar y para borrar las firmas realizadas, tuvimos que rediseñar la pantalla de firma quedando como se muestra en la **¡Error! No se encuentra el rigen de la referencia.**:

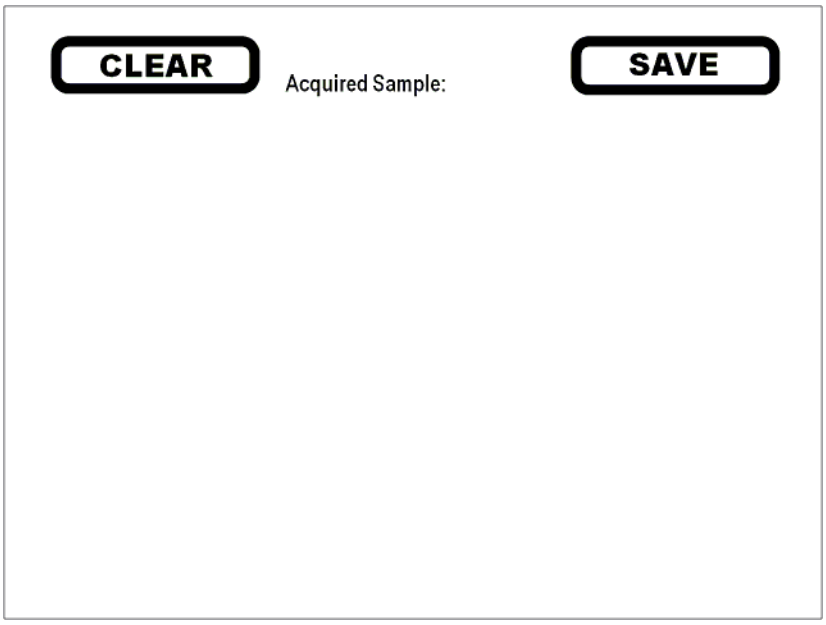

Ilustración 22. Pantalla firma Calco

Como podemos observar los botones son más pequeños y no es necesario poner un cuadro indicando al usuario donde debe firmar ya que aparecerá la imagen de la firma.

Los botones de la tableta se redimensionan como se mostró al principio del capítulo.
Surge el problema de tener que introducir dos imágenes a la vez en la tableta, por lo que nos vemos obligados buscar una alternativa. La solución encontrada fue modificar el archivo de la pantalla de calco de la figura por código de tal manera que "dibujásemos" la firma del usuario sobre el archivo de mapa de bits. De esta forma a partir de las dos imágenes obtenemos una que finalmente será la que se mostrará en la tableta. La idea es modificar el archivo de mapa de bits de la pantalla de calco "dibujando" en él la firma del usuario que queremos falsificar y finalmente formar un nuevo archivo de mapa de bits que mostraremos en la tableta.

Para poder "dibujar" sobre el archivo de mapa de bits tenemos que crearnos objetos de las clases Font, SolidBrush, PointF, Color, y finalmente un objeto de la clase Graphics.

Cuando finalizamos la adquisición de muestras del nivel 4 restablecemos los controles que teníamos antes en la aplicación, los botones de la tableta y la pantalla de firma inicial.

- Nivel 5: en este nivel tenemos que hacer uso del archivo .bir del usuario genuino, es uno de los niveles que más complicaciones ha causado a causa de su complejidad.
  En este nivel tenemos que mostrar el progreso de la firma, será el primer nivel donde se muestre de forma dinámica la firma genuina que se quiere falsificar. Únicamente se podrá visualizar una vez, al comienzo del nivel.
  El progreso de la firma se visualizará en el PictureBox del formulario inicial.
  Será necesario leer los datos del archivo guardado .bir que es el que contiene los datos sobre la firma tales como presión, tiempo y posiciones (X, Y). Por lo tanto comenzaremos explicando la lectura de datos de los archivos:

  Tenemos un método que es el GetSignature a través del cual podemos obtener los datos de las posiciones X, Y la presión y el tiempo.

  A través de este método conseguimos leer los valores de los parámetros que nos interesan, cada vez que lo llamamos y le introducimos el archivo que queremos leer obtenemos los valores deseados.

  Como podemos observar debido a que los parámetros son dinámicos (no tienen un tamaño fijo) utilizamos objetos de la clase List para almacenarlos. Los objetos de la clase List permiten el almacenamiento dinámico de la información.

Una vez que tenemos los parámetros tendremos que representarlos de tal forma que se vea el progreso de la firma. Para hacer esto la mejor solución fue crear archivos de mapa de bits en los que dibujásemos la firma basándonos en los parámetros como la presión (para dibujar el trazo) y el tiempo para controlar la velocidad, así como las posiciones X e Y. La idea fue realizar un mapa de bits en los que se dibujaran los primeros parámetros de la firma, posteriormente sustituir el mapa de bits que se está mostrando por otro en los que se han dibujado los siguientes parámetros y así sucesivamente con un tiempo de sustitución de imágenes muy pequeño permitiendo al usuario visualizar un "video" en el que se muestra el progreso de la firma.

Necesitamos un objeto de la clase Timer para controlar cuando cambian las imágenes en el PictureBox, realizaremos el cambio de imagen exactamente cada 30ms ya que si cambiamos las imágenes en ese intervalo como mínimo el usuario no se da cuenta del cambio de imagen y ve una progresión.

Tenemos que crear el mapa de bits con un tamaño que sea adecuado a la pantalla del PictureBox en nuestro caso es de el valor del máximo valor de X y de Y dividido entre 5, de este modo se adaptará a los valores de la firma.

Debido a que la tableta STU-WACOM registra valores de posición aunque el usuario no esté ejerciendo presión necesitamos filtrar estos valores que son falsos, por lo tanto mediante un bucle if filtramos todos los valores que tengan valor de presión 0.

Necesitaremos recorrer los objetos de la clase List que contienen los valores de los parámetros de las firmas, para ello utilizaremos un bucle for que será el que se encargue de ir recorriendo la lista y podremos dibujar los parámetros correspondientes.

Como podemos observar en el código el parámetro del tiempo (t[i]) siempre que sea menor que la variable tiempo que irá aumentando de 30ms en 30ms permitirá realizar el dibujo en el mapa de bits, como vemos el color elegido para dibujar la firma es el azul (aunque se podía haber escogido otro), se escogió este color por ser un color visible y agradable para el usuario, tenemos en cuenta la presión (la cual dividimos entre 40 para que no sea muy gruesa) ,dibujaremos entre los puntos de la posición X y la posición Y correspondiente (uniremos los puntos de la posición 1 (X1, Y1) con los puntos (X2, Y2) formando una línea entre ambos que al final definirá la firma. Estos puntos irán progresando de tal forma que posteriormente dibujaremos sobre el mapa de bits anterior el punto siguiente consiguiendo la visualización de la firma.

Esto lo conseguimos aumentando el tiempo de firma, como el tiempo va aumentando cada vez que salta el evento del Timer se nos permite dibujar un tramo mayor, en los siguientes 30ms podremos dibujar más tramo de la firma, y así sucesivamente hasta que finalmente se hayan leído todos los parámetros del List.

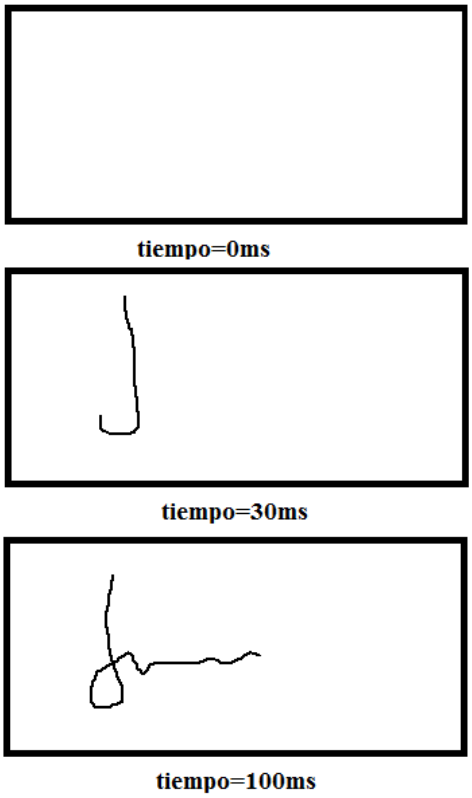

Ilustración 23. Imagen explicativa sobre el progreso de la firma

Ver Ilustración 23 para comprender mejor la solución adoptada:

Surgió un problema con los ejes de coordenadas que se registran en la tableta. Los puntos X se guardan de forma correcta, pero los puntos Y se guardan justo al contrario (el eje está cambiado) por lo que necesitamos ajustar de forma correcta el eje de coordenadas, para ello cogeremos el máximo valor de la coordenada Y de las que están guardadas en el List y le restamos el valor actual del List de esa forma conseguimos un corregir el eje de coordenadas y dibujar de forma correcta.

- Nivel 6: cuando pasamos del nivel 5 al nivel 6 restablecemos todos los valores de los List para volver a empezar a dibujar. En este nivel se consigue un control total sobre la visualización del progreso de la firma, podremos ver su progreso a una velocidad real, a menor velocidad, a mayor velocidad, parar el progreso, etc. El sistema para realizar el progreso de la firma en la pantalla del PictureBox a partir de los parámetros del archivo .bir es exactamente el mismo que para el nivel anterior, nos hemos basado en el mismo sistema.

Solo que ahora para conseguir controlar el progreso tenemos que controlar el tiempo, de tal forma que para "rebobinar" el progreso de la firma (mostrarla hacia atrás) tendremos que ir acortando el tiempo respecto al cual podremos dibujar, de esa forma cada vez se podrá dibujar menos tramo y parecerá que se "rebobina" la firma. Para el caso de parar el progreso simplemente nos mantendremos en el mismo tiempo, es decir en lugar de progresar el tiempo lo paramos, destacamos que en este punto se sale del bucle for para no colapsar el equipo. Para mostrar el progreso más deprisa se permite dibujar durante tramos más amplios de tiempo (en lugar de cada 30ms cada 60ms). Y para mostrar el progreso más lento se hace lo contrario. Para el progreso normal "Play" se utiliza un intervalo de 30ms como el utilizado en el apartado anterior.

Para la interfaz con el usuario y que por lo tanto este pueda controlar la visualización se han diseñado unos botones con formas de reproductor para que el usuario se enfrente a cosas conocidas y la interacción resulte más fácil.

Ilustración 24. Reproductor de firmas

En la Ilustración 24 se muestran los botones del reproductor de firmas.

A través de estos botones el usuario podrá visualizar en el PictureBox la firma de forma dinámica.

- Nivel 7: el diseño de este nivel se ha realizado de la misma forma que los niveles 4 y 6 juntos, se ha aplicado el mismo criterio y el mismo método para realizarlo ya que este nivel implementa las funcionalidades de los dos niveles juntos, permitirá al usuario calcar la firma y a la vez visualizar su progreso, combinamos la firma dinámica con la firma estática.

6.5. Ventanas de aviso y orientación

A lo largo de la aplicación aparecen ventanas de aviso para orientar al usuario sobre cómo debe utilizar la aplicación y en el caso de realizar alguna acción incorrecta, estos mensajes son MessageBox, elementos que permiten mostrar un mensaje al usuario.

Al haber implementado la aplicación en ingles necesitamos de una librería para que estos mensajes aparezcan en inglés, por lo que implementamos la librería MessageBoxManager.dll que permite que estos elementos aparezcan en inglés.

6.6. Desarrollo del Form1 principal.

Se decidió poner el control SignControlWacomSTU [18] en modo no visible para que el usuario no tenga distracciones a la hora de realizar las muestras, se pretende que el usuario se centre únicamente en la realización de las firmas en la tableta por lo que se ha tratado de eliminar cualquier posible distracción con la aplicación.

También hay que destacar que no se mezclan los elementos de la aplicación de captura de usuarios genuinos con los usuarios de la captura de firmas falsas. Es decir si un usuario está registrado como usuario genuino no se verán los elementos que componen la captura de firmas falsas, tales como los niveles de falsificación y los botones de reproducción de firmas.

El botón Save de la aplicación en el formulario tampoco estará visible ya que únicamente se podrá guardar mediante los botones definidos en la tableta.

La aplicación está definida de tal manera que únicamente aparecerán los controles cuando sean necesarios, también desaparecerán cuando ya no lo sean.

Esto lo conseguimos con la propiedad Visible de cada componente, simplemente activándola o desactivándola cuando queramos llegamos a esta solución.

6.7. Desarrollo del botón Stop

La función del botón Stop ya se ha explicado anteriormente, al pulsarlo se sale del modo en el que nos encontramos y volvemos al estado inicial del programa. Por lo que al pulsarlo se deben poner en no visible todos los controles que no estuvieran al inicio del programa, también en el caso de existir una imagen en el PictureBox se quitaría y se sustituiría por un mapa de bits en blanco. Y la imagen de la tableta pasará a ser la imagen de presentación que hemos diseñado. Se eliminarán todos los elementos del ListBox y el contador de muestras pasará a estar en modo no visible.

6.8. Desarrollo del formulario Help

En este formulario lo único que se ha hecho es que al pulsar el botón Help salta el evento click y da lugar a que aparezca el formulario Help (de ayuda), al que hemos llamado Form6. Este formulario únicamente se compone de un Label en el que hemos puesto el texto explicativo, este texto no se podrá modificar. Y un botón de Exit cuya única función es salir del formulario. También incluye un botón que permite imprimir una hoja de información para los usuarios “Print information sheet” al pulsarla se imprime desde este formulario un archivo .txt que hemos incluido en los recursos de la aplicación.

6.9. Desarrollo del formulario Settings

La función de este formulario es configurar por el usuario el número de muestras que el usuario quiere establecer para la aplicación de captura de firmas genuinas y para la aplicación de firmas falsificadas.

A este formulario se accede a través del botón del menú principal Settings, cuando el usuario lo pulsa salta el evento que da lugar a que aparezca el formulario Form8. Este formulario se compone de dos TextBox en los que el usuario podrá introducir los números de muestras que desee. Por defecto están configuradas en 10 muestras genuinas y 5 muestras por nivel de falsificación.

De estos datos que el usuario introduce o bien están por defecto el Form1 tomará como valores para realizar las muestras los datos que aparezcan en dichos TextBox.

6.10. Desarrollo de la interfaz de la tableta STU-WACOM

Se diseñó una pantalla de inicio para la tableta, que se activará en todo momento que la tableta no esté operativa porque no sea necesario. Tiene carácter informativo únicamente (Ilustración 25):

Ilustración 25. Pantalla presentación tableta

Fue necesario definir una pantalla de interfaz entre el usuario y la aplicación en la que se indicara al usuario donde debe firmar y donde se sitúan los botones de la tableta para aceptar y cancelar la firma, esta pantalla de firma es un archivo de mapa de bits .bmp que creamos mediante el programa Power Point, y que luego presentamos en la tableta.

La imagen que introducimos en la tableta (Ilustración 26) será agregada a los recursos del sistema de tal forma que podremos acceder a ella por código siempre que lo necesitemos aplicando las instrucciones correspondientes. Y mediante los métodos que pertenecen al signControlWacomSTU [18] podemos implementar la imagen que queramos en la tableta STU-WACOM.

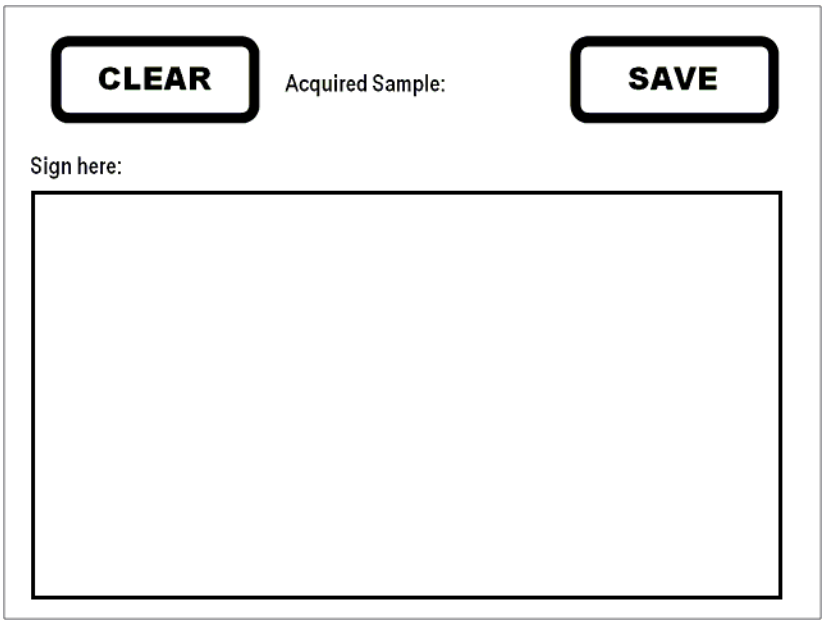

Ilustración 26. Pantalla de firma de la tableta

Una vez que está definida la pantalla donde los usuarios pueden firmar es necesario definir las posiciones de los botones "SAVE" y "CLEAR" de la pantalla. Esto se realizará mediante instrucciones de código en las que definiremos las posiciones exactas de dichos controles. Concretamente las posiciones del botón Aceptar serán (15,27) y (200,80) de la pantalla de la tableta. Y las posiciones del botón Cancelar serán (460, 27) y (660,80). Estas posiciones se han calculado de forma empírica. Estos botones también están definidos mediante el control signControlWacomSTU [18].

Una vez definidos los botones aceptar y cancelar de la pantalla de la tableta es necesario controlar los eventos para cuando el usuario pulse dichos botones se guarde la firma o se borre. Para ello tenemos que crearnos un control en el formulario que sea un button al que llamaremos Save, en los eventos del control del botón Save especificaremos que al pulsar sobre el botón llamaremos a la función del signControlWacomSTU getSignature que es la que se encarga de obtener los datos de la firma y la imagen que se genera en la pantalla del control (de esta forma conseguiremos la firma estática y la firma dinámica). De la misma forma especificaremos que si no se ha realizado la firma y se pulsa el control salte un mensaje diciéndonos que no se ha realizado la firma.

Para simplificar la interacción entre el sistema y el usuario se mostrarán el número de muestras que el usuario lleve realizadas en la pantalla de la tableta STU-WACOM, esto se ha realizado modificando el mapa de bits que se presenta en la pantalla "dibujando"

el objeto tipo (int) que lleva el número de muestras con los mismos métodos que se han aplicado anteriormente para casos similares.

De la misma forma también se mostrará el número de muestras en la pantalla del ordenador. Se mostrarán en un TextBox que tendrá asignado el objeto entero que lleva la cuenta de las muestras.

6.11 Desarrollo de la forma de guardar los archivos en el sistema

La ubicación de los archivos es fija, es decir se guardarán en una carpeta perteneciente al proyecto llamada Firmas genuinas, posteriormente cuando se asigna al usuario su número se crea automáticamente un nuevo directorio con su número, dentro de esta carpeta se guardaran sus tres archivos .txt (con sus datos), .bir (con los parámetros característicos de su firma) y .bmp (con la imagen de su firma). Los archivos se nombrarán con su número de usuario y como explicamos anteriormente el número de muestra será variable, será un contador que se irá incrementando según avancemos en las muestras, este número de muestra también se asignará al nombre de los archivos permitiendo diferenciar unas muestras de otras.

Para el caso de firmas falsificadas el procedimiento a seguir es el mismo, en este caso la ubicación para guardar estos archivos será en la carpeta Firmas Falsas, en este caso también se guardarán los archivos .bir y .bmp así como los archivos .txt correspondientes a cada usuario en un directorio creado en función a su número de usuario en el sistema, de esta forma podremos identificarles fácilmente. En este caso los archivos que se guarden en el sistema tendrán la identificación del número de usuario que ha sido falsificado, número de usuario que ha realizado la falsificación y finalmente el número de muestra.

Tanto la imagen de la firma como los datos serán almacenados temporalmente en un objeto tipo Byte[].

Ya que este es el evento donde se llevará a cabo la captura de datos y donde se crearán los archivos .bmp de la imagen de la firma en el control y .bir con los datos de la firma que se encuentran guardados en el objeto Byte es necesario llevar a cabo un control del nombre de los archivos que se están guardando para no sobrescribir las muestras. Tanto la ubicación como el nombre de los archivos vienen definidos previamente por lo que el usuario no tendrá que seleccionar ni ubicación ni nombre para guardar los dos archivos que se van a generar.

Para comparar si los archivos que se están creando tienen el mismo nombre que los archivos ya existentes se incluye en una condición por código que si el archivo que se está creando existe se incremente un contador que irá implícito en el nombre de los

archivos, de tal forma que si existe el archivo el contador se incrementará en una unidad cambiando de esta forma el nombre del archivo y pudiendo guardar una nueva muestra.

Una vez hemos definido como se guardan en el evento del botón creado los archivos de firma se relacionará el evento de botonAceptar de la STU-WACOM con el evento click del botón Save del formulario. De esta forma cuando el usuario pulse el botón en la tableta será igual que si pulsa el botón Save en la pantalla de la aplicación en el formulario (ocurrirá el mismo evento).

Hay que destacar que existen diferentes formas de guardar los dos archivos. En el código diferenciamos a la hora de guardar los archivos para usuarios genuinos existentes y usuarios genuinos recién registrados, así como los usuario Falsificadores ya existentes y nuevos. Esto lo conseguimos combinando booleanos que activarán y desactivarán las instrucciones que harán que se guarden los archivos de una forma u otra permitiendo que el usuario no se tenga que preocupar de guardar los archivos o de seleccionar una ubicación.

7 Pruebas y evaluación de usabilidad

7.1. Pruebas realizadas durante la programación

Durante el desarrollo de la aplicación se realizaron multitud de pruebas a medida que se avanzaba en la programación para comprobar que lo que estábamos implementando se realizaba de manera correcta. Gracias a estas pruebas se pudieron corregir errores en la aplicación, así como la realización de cambios en la interfaz con el usuario e incluso al encontrarnos con dificultades se decidió cambiar en muchos casos el diseño de la misma.

7.2. Evaluación de usabilidad

Debido a que éramos nosotros mismos los que estábamos realizando las pruebas en la aplicación estas podían ser no fiables, ya que al ser nosotros los desarrolladores podríamos dar por hecho que la aplicación tiene una buena usabilidad e interacción con el usuario y podría no ser así. Por este motivo se decidió realizar una evaluación de usabilidad en la que usuarios reales evaluasen nuestra aplicación, y de esta forma poder corregir los posibles fallos que surgieran.

Al hablar de usabilidades nos referimos a la cualidad de un sistema con respecto a

- Su facilidad de uso: múltiples formas de intercambiar información entre el usuario y el sistema.
- Su facilidad de aprendizaje para nuevos usuarios que garantizan interacción efectiva y máximas prestaciones.
- La satisfacción del usuario incluyendo el soporte al usuario para garantizar las metas (robustez).

Para evaluar si la aplicación cumple con unos requisitos de usabilidad aceptables expusimos a un grupo de usuarios reales para que utilizaran el sistema.

Se han escogido tres usuarios para realizar las pruebas de usabilidad:

Dos de estos usuarios están familiarizados con la tecnología utilizada y con el desarrollo de aplicaciones.

El tercer usuario no está tan familiarizado como los dos anteriores, pero es necesario ya que esta aplicación está enfocada a todo tipo de usuarios independientemente del grado de conocimientos sobre la tecnología que tengan.

Para llevar a cabo la prueba de usabilidad el sistema dispone de una base de datos de firmas legítimas de diferentes usuarios.

A los usuarios que iban a realizar la prueba de usabilidad con el sistema se les asignaron diversas tareas: la primera de ellas era utilizar la sub-aplicación de captura de firmas genuinas, que completasen el formulario de registro y realizasen un total de 10 muestras, posteriormente debían realizar otras 10 muestras pero esta vez debían identificarse en el sistema como usuarios ya registrados, para comprobar que esta parte de la aplicación funciona correctamente.

Una vez han completado esta primera tarea, debían realizar otro registro de firma legítima utilizando esta vez una firma distinta (afortunadamente los tres usuarios hacían uso de más de un tipo de firma), con lo que pudimos comprobar que la aplicación les identificaba como ya registrados y además les mostraba la firma ya realizada.
Al igual que en el proceso anterior una vez han registrado su nueva firma deben identificarse en el sistema como usuarios ya registrados, el sistema les reconoce como usuarios existentes y les mostrará las dos firmas que tienen registradas escogiendo ellos la firma sobre la que quieren realizar más muestras.

Esta primera parte funcionó correctamente sin dar demasiados problemas.

Completada la evaluación de la aplicación de captura de firmas legítimas se les asignó a los usuarios la tarea de probar la aplicación de emulación de ataques en firma manuscrita.

En primer lugar los usuarios tuvieron que identificarse en el sistema como usuarios falsificadores, completando todos los campos requeridos, una vez completado el proceso de registro se comprobó que el número de usuario que se les había asignado en la aplicación de captura de firmas legítimas no estaba entre los disponibles para falsificación por lo que esta parte comprobamos que también funcionaba correctamente.
Una vez registrados de esta forma comenzaron a realizar las falsificaciones de los usuarios que ellos escogieron, completaron los siete niveles y finalizaron el proceso.

Se les indicó que repitieran el proceso identificándose esta vez como usuarios falsificadores ya registrados para comprobar que esta parte funcionaba correctamente y el resultado fue óptimo, pudimos comprobar que los usuarios a los que en el proceso anterior habían falsificado no aparecían esta vez como usuarios disponibles para falsificar.

En todo momento los usuarios pudieron interactuar con total libertad con el sistema, pudiendo borrar las firmas que no les parecían buenas, etc.

7.3. Proceso de evaluación

Antes de comenzar con la evaluación se le hace saber al evaluador que el propósito de la prueba es únicamente la de evaluar la aplicación dejándole claro que cualquier error que cometa durante la prueba no será culpa suya, sino del diseño de la aplicación.

Durante la evaluación el usuario nos hace saber que partes de la aplicación le resulta complicada, que parte no entiende, donde comete los errores. Es importante no ayudar al usuario ni responderle a sus preguntas inmediatamente durante el desarrollo de la prueba.

7.4. Cambios en la interfaz

En esta sección analizaremos los cambios que se realizaron en la aplicación en función de las dificultades que el usuario encontró al realizar las pruebas.

En un principio cuando mostramos el programa al usuario por primera vez la aplicación no disponía de identificación por DNI, el primer problema surgió porque el usuario no recordaba el número de usuario que se le había asignado. Por lo que se decidió implementar en el programa la alternativa de búsqueda por DNI.

También encontramos el problema en el momento en el que el usuario nos indicó que tenía diferentes firmas y que no sabía cual utilizar en el registro. Por este motivo se incluyó en la aplicación de tener diferentes firmas de un usuario asignadas a diferentes números de usuario.

También tuvimos el problema de que la mayoría de los controles que se utilizaban para la aplicación de emulación de ataques de firma manuscrita estaban en modo visible por lo que el usuario en muchas ocasiones se encontraba confuso, por ello se decidió poner todos los controles que no se utilicen en modo no visible.

Encontramos el problema a la hora de realizar las muestras de que el usuario no sabía cuando detenerse, por lo que decidimos poner esto en modo automático, que fuera el programa quien le indicara cuando había terminado el proceso.

En la parte de emulación de ataques de firma nos dimos cuenta de que el usuario podía llegar a falsificarse a sí mismo por lo que se decidió eliminar la firma genuina de los usuarios falsificadores que estuvieran también registrados como usuarios genuinos en su proceso. También fue necesario incluir un contador de muestras ya que el usuario no sabía cuántas muestras estaba realizando y le provocaba incertidumbre.

La aplicación al principio no se adaptaba correctamente a las pantallas de todos los ordenadores por lo que fue necesario reducir su tamaño considerablemente.

En muchas ocasiones el usuario se sentía perdido y no sabía cómo continuar, por lo que se decidió que cuando fuera necesario que el usuario realizara otro pasó en la aplicación el programa mostrara un mensaje indicándole que pasos debía seguir.

Una vez implementados todos estos cambios obtuvimos la versión final que tiene una mejor usabilidad.

8 Creación de [1] BBDD de usuarios genuinos

8.1 Motivación

Como se ha comentado anteriormente para poder realizar el proceso de emulación de ataques de firma manuscrita y falsificar muestras de firmas legítimas era necesario disponer de una base de datos de firmas de usuarios reales.

Por este motivo se decidió realizar la base de datos.

8.2 Características de los usuarios

Se decidió reclutar a todo tipo de usuarios para hacer la base de datos debido a que así estaríamos contando con una situación más real y podríamos poner a prueba nuestra aplicación de forma más eficiente ya que al fin y al cabo se pretendió que la aplicación fuera destinada para todo tipo de usuarios.

Podemos caracterizar a los usuarios según la edad de los mismos, sexo, estudios, situación geográfica, profesión, etc. ya que estos datos pueden resultar interesantes a la hora de realizar la investigación.

La mayoría de los usuarios reclutados resultaron ser estudiantes universitarios ya que eran los usuarios más accesibles, la mayoría oscilaba entre los 18 y los 28 años.

El resto de usuarios presentaba edades más diversas que hemos clasificado en mayores de 28 años y ejercían diversos tipos de profesiones: sector de servicios, ingeniería, educación, agricultura, etc.

Todos los usuarios reclutados residen o bien en la comunidad de Madrid o de Castilla La Mancha estos datos permitirán llevar a cabo un estudio teniendo en cuenta la situación geográfica de los mismos.

Para realizar el proceso de reclutamiento se les permitió a los usuarios que realizaran su firma de la forma más cómoda y natural posible, permitiéndoles firmar de pie, sentados y en la posición que ellos escogieran o que les pareciera más cómoda, dado que intentamos en todo momento adaptarnos a las necesidades del usuario.

En la Tabla 3 Tipos de usurarios reclutados podemos observar las características principales de los usuarios reclutados:

Usuarios	Porcentaje (%)
Hombres	65.38
Mujeres	34.62
Con estudios universitarios	61.53
18-28 años	66.3
Más de 28 años	33.7

Tabla 3 Tipos de usurarios reclutados

8.2 Número de datos conseguidos

Se ha reclutado a un total de 37 usuarios, cada uno de estos usuarios tuvo que realizar un total de tres sesiones para completar el proceso de reclutamiento, en la primera sesión se le indicó al usuario que realizase 30 muestras de su firma, hay que indicar que esto en la mayoría de los casos resultó tedioso e incómodo para el usuario, pero lo que se pretendía era que el usuario automatizara la realización de la firma y finalmente obtuviéramos una muestra más real o natural de su firma.

Posteriormente se le pidió a los usuarios que realizasen otra sesión con al menos 15 dias de diferencia, el objetivo de esta diferencia de días es que el usuario olvidara el primer proceso de reclutamiento y así obtener datos más reales, además al tener tantos días de diferencia podíamos obtener muestras realizadas por el usuario con diferentes estados de ánimo.

Finalmente se le pedia al usuario una tercera y última sesión, también con una diferencia de días de al menos 15 entre sesión y sesión.

Por lo que se consiguió un total de 2220 muestras de firmas genuinas.

8.3 Organización de almacenamiento de datos

Como podemos observar tenemos una gran cantidad de muestras por lo que será necesario clasificarlas según usuario y muestra obtenida, para tener una mayor organización y llevar a cabo el estudio de forma satisfactoria.

La aplicación como hemos comentado anteriormente clasifica los usuarios en directorios automáticamente según sean genuinos o falsificadores, en este caso todas las muestras se han guardado en los directorios de usuarios genuinos, identificando cada usuario según el número que se le ha asignado en una carpeta con su mismo número, en esa carpeta se guardarán los archivos correspondientes a las muestras de las firmas que está realzando. Estos archivos se identificarán según su número de usuario y numero de muestra, de tal forma que estarán completamente identificados (ver Ilustración 27).

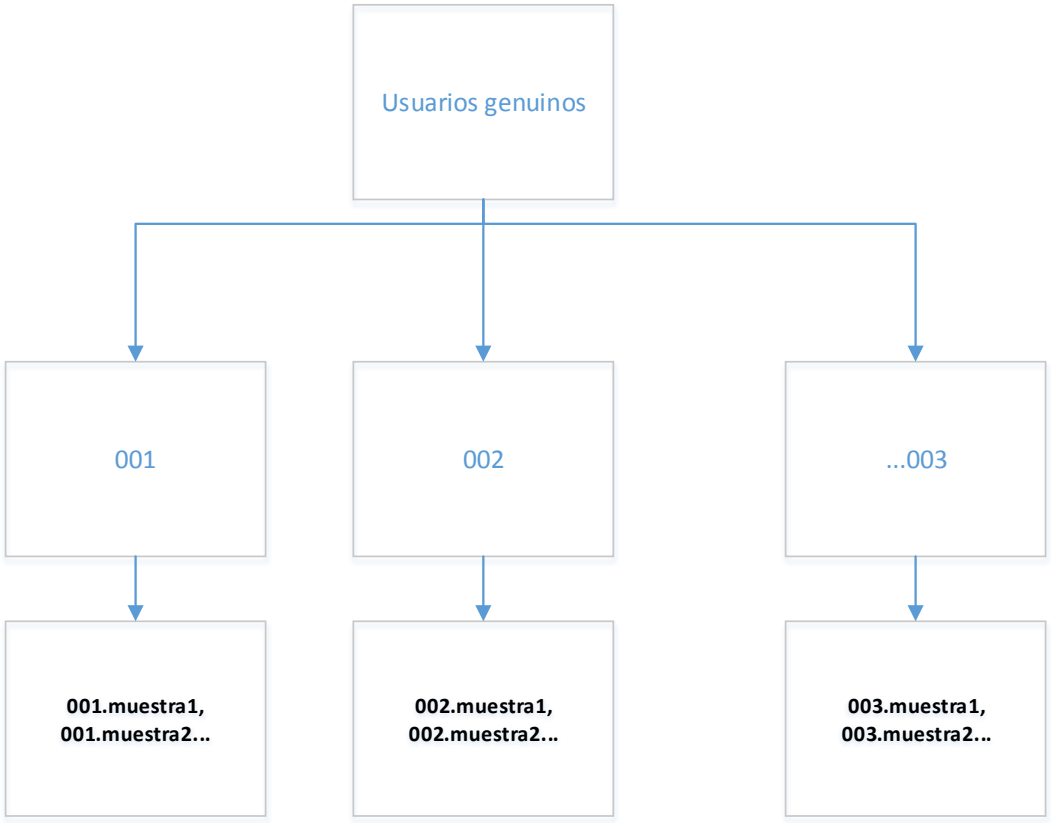

Ilustración 27 Guardar muestras

9 Conclusiones y líneas futuras

Este capítulo tratará sobre la opinión personal del autor sobre el proyecto además de comentar posibles mejoras futuras que harán más completa la aplicación.

9.1 Conclusiones

A modo de conclusión después de todos estos meses de desarrollo de la aplicación considero que es importante resaltar todo lo aprendido.

El desarrollo de esta aplicación me ha servido para comprender mejor los sistemas de reconocimiento biométrico y sus aplicaciones en las tecnologías actuales. Aprendiendo que cada aplicación requiere de un sistema biométrico que se adapte a sus necesidades, por lo que nunca podemos descartar ninguno, simplemente tendremos que seleccionar el más adecuado.

En el aspecto académico he aprendido a desenvolverme en el entorno de programación de Visual Studio, además de los conocimientos adquiridos en el entorno de la programación orientada a objetos con el lenguaje de programación C#, herramientas que podré necesitar en un futuro y que con seguridad la experiencia adquirida me será de gran utilidad. Después de estos meses desarrollando la aplicación he adquirido los conocimientos suficientes como para desarrollar aplicaciones de cierta envergadura.

Podemos concluir que la aplicación desarrollada servirá como apoyo en la investigación sobre el reconocimiento biométrico basado en firma manuscrita, proporcionando información sobre cómo puede ser falsificada una firma en función del grado de conocimiento que el falsificador tenga sobre la misma. De esta forma se podrá evaluar cómo de vulnerable es este sistema de reconocimiento biométrico, dependiendo del conocimiento que se tenga sobre la firma que se desea falsificar.

Proporciona también una herramienta de captura de muestras de firmas para una base de datos.

Espero que esta aplicación finalmente sea útil para el progreso de la investigación en las técnicas de reconocimiento biométrico basado en firma manuscrita.

9.2 Mejoras futuras

Puedo decir que estoy bastante satisfecho con el desarrollo de la aplicación pero no descarto que aún se puedan incluir mejoras que perfeccionen la aplicación. Ahora que el proyecto está finalizado se pueden comentar las funcionalidades que mejorarían la aplicación:

Uno de los primeros puntos a mejorar de la aplicación es sin duda la simplificación del código aplicado simplificando la aplicación y mejorando el rendimiento del sistema, ahora que el diseño de la aplicación está concluido posiblemente se podría haber rediseñado la arquitectura del código simplificándolo y haciéndolo más eficiente.

Bibliografía

[1] **Wikipedia - Microsoft Visual Studio.** **http://es.wikipedia.org/wiki/Microsoft_Visual_Studio**

[2] **SIC-ÁGORA - identificación biométrica y su unión con las tarjetas inteligentes.**

[3] **Dr. Raúl Sánchez Reíllo- Material docente -Tarjetas inteligentes y biometría: Identificación biométrica.**

[4] **Dr. Raúl Sánchez Reíllo-Material docente -Tarjetas inteligentes y biometría: T5-B-Áreas de conocimiento.**

[5] **Dr. Raúl Sánchez Reíllo-Material docente -Tarjetas inteligentes y biometría: ejemplo de tecnologías biométricas.**

[6] **Tabletas digitalizadoras de firmas:** **http://www.wacom.com/global-sites?**

[7] **MSDN - plataformas y herramientas de desarrollo:** **http://msdn.microsoft.com/es-es/default.aspx**

[8] **Wikipedia - Historia de Visual Studio:** **http://es.wikipedia.org/wiki/Microsoft_Visual_Studio#Historia**

[9] **MSDN - Recursos de Visual C#:** **http://msdn.microsoft.com/es-es/vstudio/hh341490**

[10] **MSDN – Introducción a Visual C#:** **http://msdn.microsoft.com/library/vstudio/a72418yk**

[11] **MSDN – Desarrollo de aplicaciones en Visual Studio**: **http://msdn.microsoft.com/es-es/library/vstudio/h8w79z10(v=vs.100).aspx**

[12] **Ceballos Sierra, Francisco Javier – Programación orientada a objetos con C++, 4ª Edición, Ra-Ma, D.L. 2007.**

[13] **Wikipedia – Dispositivos de pantalla táctil:** **http://es.wikipedia.org/wiki/Pantalla_t%C3%A1ctil**

[14] **ISO/IEC JTC 1/SC 37 N 2239 BIOMETRICS**

[15] **Wikipedia- Csharp-** http://es.wikipedia.org/wiki/C_Sharp

[16] **Grupo Universitario de Tecnologías de Identificación (GUTI) UC3M Propiedad intelectual- FullFormatSignature19794**

[17] **Grupo Universitario de Tecnologías de Identificación (GUTI) UC3M Propiedad intelectual- SignCaptureWacomSTU**

[18] **Grupo Universitario de Tecnologías de Identificación (GUTI) UC3M Propiedad intelectual- SignControlWacomSTU**

ANEXO 1. Presupuesto y planificación del trabajo

A continuación se va a llevar a cabo un desglose de las tareas que se han realizado a lo largo de este trabajo fin de grado, lo que facilitará posteriormente un cálculo aproximado sobre su coste.

Debido a la complejidad de un trabajo de estas características se ha optado por dividirlo en distintas fases, las cuales se van a comentar a continuación:

Fase 1: Documentación inicial

I. Estudio de la plataforma y del entorno de desarrollo de Microsoft Visual Studio (20 horas)
II. Preparación de las herramientas de trabajo (4 horas)
III. Búsqueda y realización de tutoriales y aplicaciones sencillas. (30 horas)
IV. Asistencia a charlas y presentaciones sobre técnicas de identificación Biométrica (16 horas)

Fase 2: Desarrollo de la aplicación

I. Actividad principal (60 horas)
II. Pantalla de configuración (20 horas)
III. Actividades secundarias (10 horas)
IV. Interconexión de todas las actividades (30 horas)

Fase 3: Pruebas

I. Pruebas genéricas de la aplicación (10 horas)
II. Corrección y depuración (20 horas)
III. Creación de [1] BBDD (20 horas)

Fase 4: Elaboración de la memoria

I. Redacción de la memoria (65 horas)
II. Corrección y maquetación (15 horas)

Tabla 4: Desglose de tareas

FASES	HORAS EMPLEADAS
Documentación inicial	70
Desarrollo de la aplicación	120
Pruebas en un dispositivo real	50
Elaboración de la memoria	80
TOTAL	**320**

PRESUPUESTO DEL TRABAJO FIN DE GRADO

COSTES MATERIALES

Los materiales necesarios han sido un ordenador, se recomienda de altas prestaciones para un correcto funcionamiento del emulador, y un dispositivo de captura de firmas digitales STU-WACOM. Considerando un periodo de amortización de cada uno de los dos dispositivos de 3 años y teniendo en cuenta el tiempo del proyecto, los costes materiales quedan como se expone en la Tabla 2.

Tabla 5: Costes materiales

CONCEPTO	PRECIO (€)
Ordenador de altas prestaciones	300
STU-WACOM	100
Licencias	200
TOTAL	**600**

COSTES DE PERSONAL

Para la realización de este trabajo, ha sido necesaria la presencia de un jefe de proyecto y un ingeniero.

Tabla 6: Costes de personal

OCUPACIÓN	HORAS	PRECIO/HORA	IMPORTE (€)
Jefe de proyecto	25	90	2250
Ingeniero	295	60	17700
TOTAL	**320**		**19950**

COSTES TOTALES

Tabla 7: Costes totales

CONCEPTO	PRECIO (€)
Costes de materiales	600
Costes de personal	19950
Costes indirectos (20%)	4110
Subtotal	24660
IVA (21%)	5178.6
TOTAL	**29838.6**

El coste total del proyecto es de *veintinueve mil ochocientos treinta y ocho euros con seis céntimos.*

Leganés, 24 de Junio de 2013

El ingeniero

ANEXO 2. Hoja de información de usuarios

Esta hoja será presentada en inglés.

USER INFORMATION SHEET

Usability handwritten signature on mobile devices
It is being invited / yy take part in a research study that aims to one side to create a database of signatures and pictures of real users and on the other create a database of signatures and pictures of users who have tried to falsify the signatures from real users applying different levels of the application.
In this way, you create two databases composed of signatures and images that will be used to carry out the study.
For the database of real user handwritten signatures is important that firms that are made are as close as possible to those that represent on paper.
For database composed of forged signatures of actual users simply follow the instructions on the levels of counterfeiting.
Before deciding to participate in this evaluation, it is important that you understand why you are conducting this data collection and what that entails. Please take time to read the following information and discuss it with others if needed. Do not hesitate to ask the person carrying out the investigation if you find something that is not clear enough or you need more information.
Purpose of the study
Our intention is to simulate environments similar to those that the user can be found usually in order to test the usability of the device used in this case (WACOM) at the time of collecting handwritten signatures. The handwritten signature is a biometric modality by which it is possible to identify people uniquely through its signature. With this database, in addition to measure usability, helping to improve biometric recognition systems for handwritten signature.
The study that we carry out is about how users interact with the system, what kind of difficulties they are and how to avoid them.
We would like you completase three visits that would occur with at least one week apart. At each visit, you would have to sign on several occasions on the device to use (WACOM). An operator will explain the process to start and will assist in the evaluation to answer your questions. The same operator will indicate when the evaluation is completed.
This study consists of three visits in which the user will complete the same scenarios:

• Visit 1: They will explain to you the meaning of the evaluation and will be given this form of acceptance. Then start collecting signatures.

• Visit 2: No explanation is necessary the system and go directly to the collection of signatures.

• Visit 3: Similarly, no explanation is necessary the system and go directly to the collection of signatures. At the end of the visit, you will be given a brief satisfaction questionnaire to tell us your opinion on the evaluation.

The time spent in each session depends on the user, as there may be times of rest when the user sees fit.

The system used is absolutely secure. It will happen no physical damage during the evaluation.

What about firms that do?

All information you provide will be part of a database belonging to GUTI (University Group for Identification Technologies), which is part of the Department of Electronics at the University Carlos III of Madrid. This information will only be used by the University Carlos III and will be only for research purposes.

The collected signatures will be stored with a user number instead of name. Only the research team collected data will be able to relate their signatures with your personal information and this information will be kept confidential within the research group.

What will happen to the results of the evaluation?

The results of the evaluation will be documented and submitted for publication to the scientific community in order to improve the state of the art. However, no participant will be identified individually and no shows will be published without prior permission.

In compliance with the Organic Law 15/1999 of December 13, Protection of Personal Data (LOPD) is informed that the data collected through this form will be included in a file of personal data held by the Carlos III University of Madrid. Your data will be protected in accordance with current legislation, will be treated confidentially and with the utmost privacy, with the purpose for which it was supplied and in order to keep you regularly informed of our activities and services. You can exercise your right of access, rectification, cancellation or opposition to treatment, sending a letter to that effect, along with a photocopy of your ID to the following address: Calle Madrid, 126, 28903 Getafe, Madrid.